

The Auto-Board System

Schematic Capture
PC Board Layout
Auto-Routing



020-1

\$175 } \$225
\$.50

Copyright ©1986 - The Great SoftWestern Company, Inc.
All rights reserved

This publication, or parts thereof, may not be reproduced in any form, by any method, for any purpose. The Great SoftWestern Co., Inc. makes no warranty, either expressed or implied, including but not limited to any implied warranties of merchantability or fitness for a particular purpose, regarding these materials and makes such materials available solely on an "as is" basis. The Great SoftWestern Co., Inc. shall not be liable to any person for damages in connection with or arising out of purchase or use of these materials. The sole and exclusive liability to The Great SoftWestern Co., Inc., regardless of the form of action, shall not extend the purchase price of the materials described herein.

The Great SoftWestern Company, Inc.

August 17, 1986

The Great SoftWestern Co., Inc. reserves the right to improve its products as it sees fit. This publication describes the state of the product at the time of its publication, and may not reflect the product at all times in the future. This manual was prepared and published on August 17, 1986 and is based on Version 2.0 of The Auto-Board System II.

IBM and PC-DOS are registered trademarks of International Business Machines Corporation. MS-DOS is a registered trademark of Microsoft Corporation. Auto-CAD, ADE-1, ADE-2 and AutoLISP are registered trademarks of Autodesk, Inc.

This manual was prepared using T_X and L^AT_EX with the P_{OT}X system from Personal T_X, Inc. It was typeset on a Hewlett-Packard LaserJet printer with the



Copyright ©1986 – The Great SoftWestern Company, Inc.
All rights reserved

This publication, or parts thereof, may not be reproduced in any form, by any method, for any purpose. **The Great SoftWestern Co., Inc.**, makes no warranty, either expressed or implied, including but not limited to any implied warranties of merchantability or fitness for a particular purpose, regarding these materials and makes such materials available solely on an “as-is” basis. In no event shall **The Great SoftWestern Co., Inc.** be liable to anyone for special, collateral, incidental or consequential damages in connection with or arising out of purchase or use of these materials. The sole and exclusive liability to **The Great SoftWestern Co., Inc.**, regardless of the form of action, shall not exceed the purchase price of the materials described herein.

For condition of use and permission to use these materials for publication in other than the English language, contact **The Great SoftWestern Co., Inc.**

The Great SoftWestern Co., Inc. reserves the right to revise and improve its products as it sees fit. This publication describes the state of this product at the time of its publication, and may not reflect the product at all times in the future. This manual was prepared and published on August 17, 1986 and is based on Version 2.0 of **The Auto-Board System IItm**.

IBM and PC-DOS are registered trademarks of International Business Machines Corporation. MS-DOS is a registered trademark of Microsoft Corporation. AutoCAD, ADE-2, ADE-3 and AutoLISP are registered trademarks of Autodesk, Inc.

This manual was prepared using $\text{T}_{\text{E}}\text{X}$ and $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ with the $\text{PCT}_{\text{E}}\text{X}$ system from Personal $\text{T}_{\text{E}}\text{X}$, Inc. It was typeset on a Hewlett-Packard LaserJet+ printer with the Tall Tree J laser adapter.

Preface

The Auto-Board System II program is a computer aided design (CAD) tool to assist the electronic engineer or technician in the design, layout and construction of printed circuit boards (PCB). Unlike some printed circuit CAD systems which require the speed and resources of the mainframe computer, The Auto-Board System II is designed to run in the environment of the micro-computer. This makes the PCB workstation concept available to individual engineers and technicians at a reasonable cost; whether they are using a micro-computer at their desk in a large corporation or at a small firm or home office as an independent consulting engineer.

The Auto-Board System II is companion software to AutoCADTM. One of the reasons for the success of AutoCAD is the existence of software drivers for a very large field of hardware peripherals; plotters, digitizers, printers and video systems. By assuming the specialized work of printed circuit design, but relying on AutoCAD for graphically oriented human interaction, The Auto-Board System II provides dedicated working conditions to this general "graphics operating system." One benefit of this relationship is that AutoCAD continues to be a general design arena when the designer's needs turn to package or mechanical design.

The Auto-Board System II is capable of performing the following operations:

- Schematic capture
- Component placement
- Rubber-banding of ratsnest lines to aid layout
- Netlist generation
- User defined library elements
- 50 mil and fine line auto-routing
- Memory array routing
- Buss layout and routing

- Multi-layer routing
- Surface Mounted Technology
- "Hooks" for future toolkit development
- Silkscreen preparation
- User defined pad shapes
- Drill coordinate and size listings

The Auto-Board System II has few required user interactions. It gathers its information from AutoCAD files and component databases. It delivers the results of its operations to AutoCAD through script files. We could get into all kinds of discussions about the logic for this approach; but, so far, a better alternative has not been forthcoming (inserting user defined parts would be extremely difficult for both of us with DXF files).

AutoCAD is used for the following facilities:

- Movement of traces to allow insertion of "missed" traces
- Further optimization of traces
- Production of check plots
- Insertion of text, special symbols and targets
- Insertion of filled pads for final plots
- Production of DXF files for post-processors
- Standardization of drawings

This user's guide documents all of the functions of **The Auto-Board System II** Version 2.0. To the best of our knowledge, there are no known "disasterous" bugs in the software. However, there is no such thing as "bug-free" software when working at this scale. It is also a fact that software takes on the personality of the author and certain things work in a fashion which is comfortable for the author. We want to hear from you with any information about **The Auto-Board System II** that you feel is important. We certainly want to know about program bugs, inconvenient conventions and/or rough spots. Be ready to be specific and to provide documentation and examples.

Write to us at:

The Great SoftWestern Company, Inc.
207 W. Hickory St. #309
Denton, Texas 76201 USA

or call us at:

Technical Support	(817) 383-4434
Order	(800) 231-6880
TLX	265792 GSWST UR
FAX	(214) 661-1096

Thank you for your interest in **The Auto-Board System II**.

Dennis N. Jump
President

This user's guide documents all of the functions of The Auto-Board System II Version 2.0. To the best of our knowledge, there are no known "disastrous" bugs in the software. However, there is no such thing as "bug-free" software when working at this scale. It is also a fact that software takes on the personality of the author and certain things work in a fashion which is comfortable for the author. We want to hear from you with any information about The Auto-Board System II that you feel is important. We certainly want to know about program bugs, inconvenient conventions and/or rough spots. Be ready to be specific and to provide documentation and examples.

Write to us at:

The Great SoftWestern Company, Inc.
307 W. Hickory St. #309
Denton, Texas 76201 USA

or call us at:

Technical Support	(817) 383-4434
Order	(800) 331-6880
TLX	365792 GSWST UR
FAX	(214) 681-1006

Thank you for your interest in The Auto-Board System II.

Dennis N. Jump
President

Contents

1	Introduction	1
1.1	Program Overview	1
1.1.1	Schematic Capture	1
1.1.2	Component Placement	3
1.1.3	Ratsnest Generation	3
1.1.4	Auto-Routing	5
1.2	System Requirements	7
1.3	User Guide Overview	7
1.4	Update Policies	9
2	Getting Set-Up	11
2.1	Introduction	11
2.2	Installation	11
2.3	Setting Up MS-DOS	14
3	Schematic Capture	19
3.1	Introduction	19
3.1.1	Before You Start	19
3.1.2	Overview	20
3.2	Using Schematic Capture	21
3.2.1	Using the Batch File	21
3.2.2	Drawing the Schematic	23

3.2.3	Special Techniques	36
3.3	Schematic Capture Software	44
3.3.1	Attribute Conversion	45
3.3.2	DXF Conversion	45
3.3.3	Netlist Generation	45
3.4	The Batch File Examined	45
4	Component Placement	49
4.1	Introduction	49
4.2	Laying Out a Board	50
4.2.1	Starting the System	50
4.2.2	Setting Up Component Placement	50
4.2.3	Pick and Place Parts	53
4.2.4	Special Techniques	56
4.3	Ending the Session	64
4.3.1	Writing the Files	65
4.4	Component Placement Software	65
4.4.1	Netlist Generation	66
4.4.2	Buss Detection and Conversion	66
4.5	The Batch File Examined	66
5	The Auto-Router	69
5.1	Introduction	69
5.2	The Batch Files	69
5.2.1	Using the RTE Batch Stream	70
5.2.2	Using the RAT Batch Stream	70
5.3	Introduction to Programs	72
5.3.1	The NET Program	72
5.3.2	The Routing Program (ROUTE)	74

5.3.3	The Configuration Files	78
5.4	Finishing a the Artwork	81
6	Custom Parts - Schematic Symbols	85
6.1	Introduction	85
6.2	Schematic Symbol Creation	86
6.3	AutoCAD Script Preparation	87
6.3.1	Initialize AutoCAD	88
6.3.2	Draw the Symbol	88
6.3.3	Symbol Attributes	90
7	Custom Parts - Layout & Router	97
7.1	Drawing the Part	97
7.2	Layout Attributes	99
7.3	Router Attributes	102
8	The Database Files	105
8.1	The Database Record	105
8.1.1	Symbol/Part Name	106
8.1.2	Pin Count	106
8.1.3	Pin Offsets	106
8.2	Build or Update the Database	108
A	The Netlist Language	111
A.1	Introduction	111
A.2	Construction of the List	112
A.2.1	Part Definition	112
A.2.2	Package Placement	114
A.2.3	Signal Name	115
A.2.4	Signal Connection	115

78	A.2.5 Device Type	116
81	A.2.6 Include File	117
82	A.2.7 Comment	117
83	A.2.8 End of List	117
86	B Router Intermediate Programs	119
87	B.1 Introduction	119
88	B.2 Array Processor	119
88	B.3 Edge Connector	120
90	B.4 Header	121
97	B.5 Buss	121
97	B.6 Signal Sorting	122
97	B.7 Signal Grouping	122
99	B.8 Signal Screening	122
102	B.9 Prerouter	123
105	C Symbol Libraries	125
105	8.1 The Database Record	125
105	8.1.1 Symbol/Part Name	125
105	8.1.2 Pin Count	125
105	8.1.3 Pin Offsets	125
108	8.2 Build or Update the Database	125
111	A The Netlist Language	125
111	A.1 Introduction	125
112	A.2 Construction of the List	125
112	A.2.1 Part Definition	125
114	A.2.2 Package Placement	125
115	A.2.3 Signal Name	125
115	A.2.4 Signal Connection	125

List of Figures

1.1	Typical Schematic Drawing	2
1.2	Typical Component Layout	4
1.3	Typical Solder Layer of Routed Board	6
3.1	Organization of Schematic Capture Module	20
3.2	Initialized Schematic Workspace	26
3.3	Orientation in Gate Symbols	29
3.4	Schematic Symbols in the Workarea	30
3.5	The "Ball"	31
3.6	Offsheet Tags	31
3.7	Bussing	32
3.8	Ground	32
3.9	Signal Terminals in the Schematic	33
3.10	Connecting Parts with Signal Lines	35
3.11	Crossing Signals Being Connected in the Schematic	37
3.12	Array - Common Connections	39
3.13	Array - Individual Connections	40
3.14	Power and Ground Dilemma	41
3.15	Power and Ground Symbol with Bypass Capacitors	42
3.16	Power and Ground Symbol	43
4.1	Board Edges Drawn	51

5.1	Router Options	82
6.1	Hand Drawn Symbol Design	87
6.2	Outline of the Schematic Symbol	89
6.3	Drawing of Schematic Symbol with Attributes	95
6.4	Schematic Symbol in a Drawing	96
7.1	Component Outlines	100

1	1.1 Typical Schematic Drawing	82
4	1.2 Typical Component Layout	87
8	1.3 Typical Solder Layer of Routed Board	89
10	2.1 Organization of Schematic Capture Module	95
16	2.2 Initialized Schematic Workspace	96
19	2.3 Orientation in Gate Symbols	100
20	2.4 Schematic Symbols in the Workspace	100
31	3.5 The "Ball"	100
31	3.6 Offset Tags	100
32	3.7 Bussing	100
32	3.8 Ground	100
33	3.9 Signal Terminals in the Schematic	100
35	3.10 Connecting Parts with Signal Lines	100
37	3.11 Crossing Signals Being Connected in the Schematic	100
39	3.12 Array - Common Connections	100
40	3.13 Array - Individual Connections	100
41	3.14 Power and Ground Dilemmas	100
42	3.15 Power and Ground Symbol with Bypass Capacitors	100
43	3.16 Power and Ground Symbol	100
51	4.1 Board Edges Drawn	100

Chapter 1

Introduction

1.1 Program Overview

The **Auto-Board System II** is a package of software tools and **AutoCAD** interfaces to aid the electronic designer with the production of printed circuit boards. The capability of routing up to 16 layers is provided in advanced versions of the package.

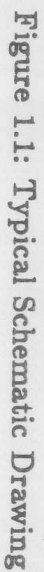
The system consists of four major software products:

- Schematic capture and netlist generation
- Component placement on the circuit board
- Ratsnest generation for layout optimization
- Auto-routing of the captured design.

Each software product handles one of the phases of circuit development. The **Auto-Board System II** is matched to **AutoCAD** which is a major micro-computer based CAD system. **AutoCAD** is the basis for all human interactions with the computer.

1.1.1 Schematic Capture

The schematic capture module is usually the first encounter with **The Auto-Board System II** that the designer has. As the designer sees the module, schematic capture is a set of schematic symbols and menus that are used to draw an **AutoCAD** schematic diagram. Schematic symbols are selected from the schematic library to form the inner structure of the drawing. This library of drawings provides the designer with



a standard group of symbols that can be manipulated with **AutoCAD** to produce a complete definition of a circuit. Connections between the individual leads of the symbols may shown as connecting lines, buss lines and "named signals." Figure 1.1 is a typical schematic diagram produced with **AutoCAD**.

This design task is performed entirely from within **AutoCAD**, allowing the designer to utilize such commands as **MOVE**, **ERASE**, **STRETCH** and **ATTEDIT** (attribute edit). Screen menus and third party products (tablet menus, extension libraries and **AutoLISP** programs) can be used to assist in the design task.

The library of symbols may be expanded to include custom parts that the user may have or encounter. This expansion is explained in chapters 6 through 8.

To **The Auto-Board System II**, schematic capture is more than a series of symbols and lines. The software portion of the package is designed to generate a netlist of the schematic from the files that are saved by **AutoCAD**. The symbols of the schematic library have been formed to convey information to the schematic capture software through the use of attributes and layer provisions of **AutoCAD**. The netlist that is produced by the schematic capture module is 100% compatible with the auto-routing software.

1.1.2 Component Placement

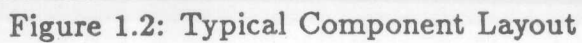
With the schematic designed, the design sequence moves to component layout. As in the schematic capture module, the designer sees the component placement or layout phase of **The Auto-Board System II** as a set of silkscreen and pad master drawings of various electronic devices and an **AutoCAD** menu. With the menu, the designer can pick and place the physical components as they are needed on the printed circuit board. There are provisions to map the outline of the board itself, declare "masked" no-entry areas and define guard fences to partially protect areas of the board. Figure 1.2 illustrates a typical layout.

To **The Auto-Board System II**, the software provides part definition and location information to the auto-router. This data is directly readable by the routing package.

1.1.3 Ratsnest Generation

To aid in the optimization of component placement, a ratsnest generating program is used to overlay the component placement drawing with straight line connections, a "ratsnest". This module is actually a blend of the component placement module and the auto-router.

Recent versions of **AutoCAD** allow components and attached ratsnest lines to be



dragged around the screen while retaining the connectivity of the lines at the stationary end. This is performed with the STRETCH command and allows the powerful concept of "rubber-banding." The sequence of events that produces the ratsnest returns the designer to the component placement phase. Any improvements that may be indicated by the ratsnest can be directly implemented, creating a new set of layout parameters.

1.1.4 Auto-Routing

The auto-router is a series of programs providing the routing of printed circuit boards. The entire system is composed of translation, transformation and routing programs, plus a library of AutoCAD drawing files for insertion into the final drawing. The drawings are silkscreen and pad master representations of the parts on the printed circuit board. Figure 1.3 shows the solder side of a routed board.

The "star" program in The Auto-Board System II is the automatic trace router which is responsible for physically connecting of nets. Its output is a stream of ASCII instructions to AutoCAD which is known as a SCRIPT file. The AutoCAD INSERT instruction is used to retrieve component templates from the library of drawings that are supplied with the system, third party sources or by the user.

The user has a great deal of control in setting the rules by which the router will operate. These rules are given to the router through configuration files (up to 3 at a time). Rules take the form of simple equations. For instance, setting a default trace width to 10 mils is accomplished by entering $tw = 10$ in the configuration file. Another command allows the user to select the type of AutoCAD trace that the router will produce. You can choose LINE, TRACE or PLINE forms of lines to draw tracks.

To keep certain portions of the drawing separated, the AutoCAD layers are used extensively. The router will define the following layers:

- PADS
- SILK
- RATSNEST
- Ln where $n = 1$ to 16
- DRILL
- SOLDER
- COMPONENT

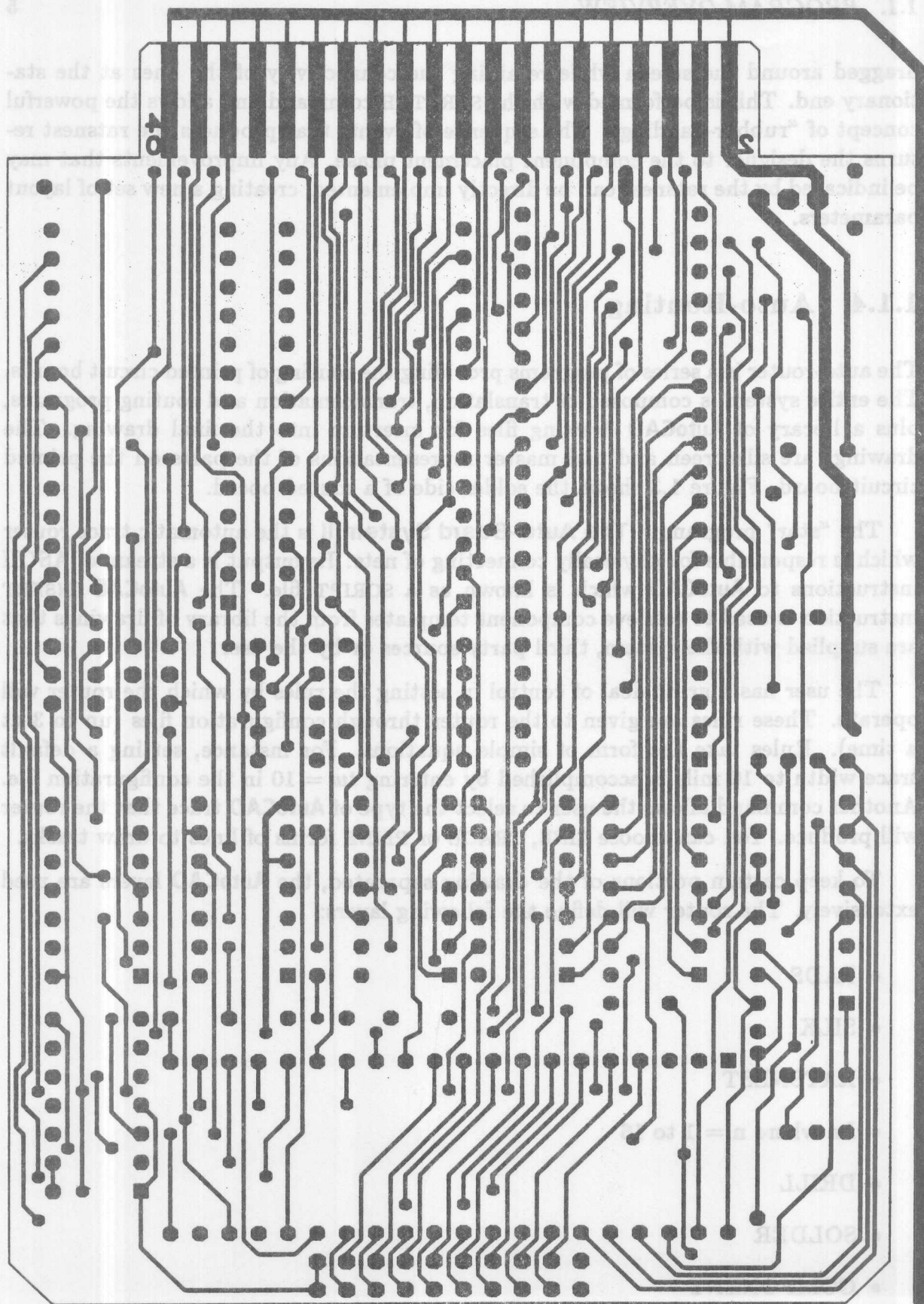


Figure 1.3: Typical Solder Layer of Routed Board

1.2. SYSTEM REQUIREMENTS

7

- PWR
- GND

The Auto-Router is capable of handling the following volume of information:

- 1000+ individual components
- 40 Bus Definitions
- Unlimited I.C. Arrays
- 1000+ Pins using the same signal
- 4000 Trace Segments (50 mil router)

The fine line auto-router is capable of working on a 16" by 16" board with 640K of memory. The 50 mil grid router will route in a 24" by 22" area.

The auto-router is written entirely in 'C' with some direct access to the MS- DOS operating system. The original routing algorithm was developed on a NEC APC with the graphics (7220 Graphic Processor) card with 1024 X 1024 resolution. This allowed us to SEE the action of the router in real-time. To hide the exact gyrations of the algorithm, the plotting has been moved to the processor RAM.

1.2 System Requirements

To use The Auto-Board System II with the IBM PC or other compatible microcomputer, you will need 640K of memory and a hard disk (10M is minimum). Any other fancy equipment (color board, monitor, plotter and digitizer) that you might need for good PCB work will be requirements of AutoCAD system. They are certainly nice to have, but are not *operational* requirements of The Auto-Board System II.

Certain functions of this system will utilize commands found in the later versions of AutoCAD. At this time, AutoCAD 2.18 can be used for schematic capture, component placement and auto-routing. The rubber-banding of ratsnest lines in the ratsnest/component placement module requires AutoCAD 2.5.

1.3 User Guide Overview

This user guide provides information about the working details of The Auto-Board System II. To get the full value from your system, you should be very familiar with

the use of **AutoCAD** and some of the features of the MS-DOSTM operating system. This manual cannot hope to be a tutorial for those products. These software products have several reference texts available (especially MS-DOS). Please refer to them for further information.

The **Auto-Board System II** User Guide covers the following information:

Chapter 1 A general overview of the system, its capabilities and its relationship to the **AutoCAD** system.

Chapter 2 The installation chapter. The emphasis of this chapter is getting the files of the distribution diskettes installed on the hard disk of your system in an organized manner. This chapter also deals with the use of MS-DOS commands.

Chapter 3 The schematic capture package. This is divided into three parts. First, the use of the schematic capture symbol library and **AutoCAD** to create a schematic drawing is explained. The second part is an overview of the schematic capture software. The third part is a technical look at the batch file.

Chapter 4 The component placement package. Following the outline of the third chapter, this chapter also explains the role of **AutoCAD** and the **LAYOUT** library to place the components of the electronic circuit, the function of the **LAYOUT** program and the use of the batch files to aid you in running the system.

Chapter 5 The auto-routing package. This chapter deals with the software with which you come in contact. These programs are: **NET** and **ROUTE**. There are several programs in the package. However, the batch files control their execution and you will hardly ever have to pay any attention to them. This chapter also deals with the batch streams used in running the auto-router.

Chapter 6 The creation of schematic library symbols.

Chapter 7 The creation of layout auto-router components.

Chapter 8 The creation of new symbols and parts and their addition to their respective databases.

Chapter 9 Techniques used to get the most out of the system. Use this chapter as a place to store ideas, exchange ideas with others and save any notes that we may send out. (Intentions are great, but it is difficult to promise newsletters, etc. for the future).

Appendix A A complete definition of the netlist language.

Appendix B A review of the intermediate programs found in the auto-router package.

Appendix C Plots of current library parts.

1.4 Update Policies

The Great SoftWestern Co., Inc. is continuously working on major and minor enhancements and upgrades to its products. These newer versions will be made available at regular intervals (sooner if necessary).

The first level of update will involve the correction of errors, user inconveniences and small program changes. Small program changes involve the inclusion of helpful changes, speed improvements and minor "design rule" modifications. For instance, one of the "design rules" being considered is the exclusion of vias under integrated circuits. Updates for this level will be announced through user newsletters and mailings to registered owners of **The Auto-Board System II**. By returning your original program diskette and a reasonable shipping and handling fee, we will update your program. Feedback from you will help us in deciding upon what modifications are desirable.

The larger update will be the introduction of a related product. When you purchase one of the systems, later versions will be made available to you at the difference in price between the newer version and the original price.

This applies only to registered users.

Chapter 2

Getting Set-Up

2.1 Introduction

This chapter covers the installation of the system programs, library symbol drawings and support files of **The Auto-Board System II**. You can think of this chapter as the chapter that deals with the interface of **The Auto-Board System II** with the MS-DOS operating system (that's redundant). Besides copying the files of the system to your hardware in an organized manner, there will be a discussion about setting up the MS-DOS environment section so that **AutoCAD** and **The Auto-Board System II** can be used in the most flexible way.

The diskettes in **The Auto-Board System II** are NOT copy protected (yet!). It is an excellent idea to make backup copies of the distribution diskettes for archival purposes. Copies of the diskette's contents are for archival purposes only and not for distribution or sale.

The Auto-Board System II can be found on a set of distribution diskettes. Past experience has shown us that manual installation of these diskettes can lead to confusion and mistakes. To eliminate these problems and to set up the system in a standard way, an installation control disk has been added to the system. Please use the installation procedure.

2.2 Installation

The installation starts with the use of the **INSTALLATION** diskette. To install **The Auto-Board System II**, follow these steps:

- Find the diskette marked **INSTALLATION**

- Insert it into floppy drive "A"
- Enter the following:

```
a:install
```

- Follow the instructions that appear on the screen

If you get a message that the installation had been stopped because a verification file could not be found, check the following items:

- Check that the *requested* disk was in the floppy drive.
- Use the *CHKDSK* program to see if the disk is damaged. Enter:

```
chkdsk a:
```

- Start the installation procedure again; taking extra care that the correct disks are used.

To assist you in using the correct disks, here is the order in which they are requested:

1. INSTALLATION
2. SOFTWEST 1
3. SOFTWEST 2
4. SCHEMATIC LIBRARY 1
5. SCHEMATIC LIBRARY 2
6. LAYOUT LIBRARY
7. ROUTER LIBRARY
8. ROUTER SOFTWARE

There are a couple of things that you must do to finish the installation of the system.

The first thing that you will have to do is to create a working directory. The **Auto-Board System II** and **AutoCAD** are both designed to work on drawings and data files in separate working directories. This is to say that you don't have to designate your ACAD directory or SOFTWEST directory as the current directory to run the programs.

To create a working directory, follow these steps:

- Think of a name to call the working directory. We'll use PCB as an example.
- Create the directory with the **Make Directory** command. Enter:

```
md \pcb
```

To use this directory in everyday activity, tell the operating system (PC-DOS or MS-DOS) that PCB is the current directory. Enter:

```
cd \pcb
```

The leading “\” in front of PCB tells MS-DOS that the directory is located in the “root” directory.

The other thing that has to be done is to make a decision. We scoured the documentation looking for a standard in numbering edge connectors and headers. We didn't find one. However, three general methods were dominate. These methods are:

1. Method “A” numbered the pins on one side of the connector consecutively and then continued to number the pins from the “far” end of the connector back to the “near” end on the other side. This is the same way that ICs are numbered. We have heard of this being called the Hairpin method. Method “D” is the mirror image of “A”.
2. Method “B” started the numbering on one side in the same manner as method “A”, but numbered the pins on the “other” side from the “near” end again, numbering consecutively to the “far” end. This may be called the Parallel method. Method “E” is the mirror image of “B”.
3. Method “C” is completely different. In this method, the odd numbered pins comprise one side of the connector while the even numbered (odd number + 1) pin is the corresponding pin in the adjacent row. This may be called the Zig-Zag method. Method “F” is the mirror image of “C”.

Your decision involves deciding which method you prefer to use. Or which method a project demands. On the distribution diskette are two sets of programs that require this decision, the edge connector (EDGECON) programs and the header (HEADER) programs. Currently, they are marked as:

- EDGECONA and HEADERA for Method A
- EDGECONB and HEADERB for Method B

- EDGECONC and HEADERC for Method C
- EDGECOND and HEADERD for Method D
- EDGECONC and HEADERE for Method E
- EDGECONF and HEADERF for Method F

All of these files are .EXE files. When you have decided which version to use, you can do several things. Three logical choices would be:

1. Copy the selected programs to

- EDGECON.EXE
- HEADER.EXE

as the programs are referred to in the batch files.

2. Change the RTE and RAT batch files to agree with the selected program. To do this, edit the RTE.BAT and RAT.BAT files and change the lines referring to

- EDGECON
- HEADER

to (for instance)

- EDGECONA
- HEADERA

3. Create a batch file (copy RTE.BAT and RAT.BAT) for each combination that you think that you will be using. For instance, you could copy RTE.BAT to RTEA.BAT and modify the batch file as shown in option 2. Do this for all types "B" and "C" also.

2.3 Setting Up MS-DOS

One of the benefits of AutoCAD 2.1+ is its ability to run correctly even though the AutoCAD program is not in the current directory. Further extensions allow secondary libraries (neither in the current directory nor in the AutoCAD directory) to be searched. This means that working drawings can be kept in a sub-directory of their own (the current directory) and they can be manipulated with AutoCAD with access to multiple libraries.

To continue with this flexibility, The Auto-Board System II is also capable of running programs and databases in other sub-directories. You therefore have access to all of the programs that you will require without having to copy drawing files, netlists and script files from one directory to another.

We will be using the MS-DOS commands, *PATH* and *SET*, to tie everything together.

The *PATH* command allows us to "be in" a current directory on the hard disk (as a data path) and activate programs and batch files in another directory to operate on data in the current directory. *PATH* allows you to define the names and paths of all of the sub-directories to be searched to find a program (.COM or .EXE) or batch (.BAT) file. *PATH* will allow us to tell the operating system where to find AutoCAD and the programs of The Auto-Board System II when the current directory contains our schematic and routed board drawings.

When we request a program to be run, MS-DOS will check the current directory for the existence of the requested program. Not finding it (there should only be drawing files, netlists and intermediate data files in the current directory), MS-DOS starts checking each of the directories named in the *PATH*. When it finds the requested program, the program is allowed to run. This program will usually operate on the data files of the current directory. *PATH* is used only for the retrieval of program files and is not used for finding data.

The main programs of interest here are The Auto-Board System II and AutoCAD. We will establish the path for these programs at this time. You may have to expand the process somewhat to allow entries for other systems in your computer.

We need to add the ACAD and SOFTWEST directories to the path.

Let's begin by finding out if there is a path already in the system. Enter:

path

With this command, you will either get a message, such as,

"No path set"

or the word "PATH" followed by a set of cryptic mumbo jumbo. We are going to add to the mumbo jumbo (or create it from scratch, if necessary). If there is no path (I really doubt it), you will need to enter the following command.

This command will set a path to the

- root directory

- the **SOFTWEST** program directory
- the **ACAD** directory

Enter:

```
path=\\;\softwest;\acad
```

You could certainly add several other directory names to the list with little problem. In the case that there is already an existing path, you will have to:

- Re-enter the entire path as it appears in the response
- Add a semi-colon
- Add the pathname of the **SOFTWEST** directory
- Add a semi-colon
- Add the pathname of the **AutoCAD** directory

Typically, the path will be set by an entry in the **AUTOEXEC.BAT** file. This file contains commands to be used whenever the system is "booted". Use a text editor (**EDLIN** in **MS-DOS**) to add the path command or extra directory reference to an existing path command in the **AUTOEXEC.BAT**.

The purpose of the **SET** command is to store information in a section of memory known as the environment section. This area is controlled by the operating system. Both **The Auto-Board System II** and **AutoCAD** recognize certain entries produced and saved by the **SET** command. Both products will make inquiries to the environment area and if you have left the proper entry, good things will happen. In the case of **AutoCAD**, the **PATH** command will allow **MS-DOS** to find the **AutoCAD** program, but the **AutoCAD** program needs to find its overlays, support files and configuration. **PATH** will not be helpful in these cases. To aid **AutoCAD** we will leave an entry about the location of the configuration file with the **SET** command. We will also be able to tell **AutoCAD** where to find a secondary library of drawings through the environment section.

There are two **AutoCAD** related entries of which you should be aware. These entries are **ACADCFG** and **ACAD**.

ACADCFG is the name of the environment entry that reports the path name of the directory that contains the configuration that you wish to use with **AutoCAD**. By setting this message to various values, you can specify that **AutoCAD** be configured

differently from one session to the next. This can mean different hardware characteristics or initial menu display. If your configuration file is in the standard **AutoCAD** directory, the entry for **ACADCFG** would be set to the **ACAD** directory. Here is the command that you would enter:

```
set acadcfg=\acad
```

The only space in this line is after the **SET** command. The value of the current directory is irrelevant. **SET** can be used from any directory at any time.

ACAD is the name of the environment entry that tells **AutoCAD** the name of any secondary library that should be searched when there is a request for a block drawing, menu, attribute template or script file. The batch files of **The Auto-Board System II** will be changing this message as your design progresses from one module to the next. For instance, the command to assign the secondary library to the **ROUTER** directory so that the final artwork can be constructed will be:

```
set acad=\router
```

Once again the only space is after the **SET** command.

You will not have to set these values if you stick to the batch files that accompany this system. However, if you should get curious or choose not to use the batch file (I don't know why, but this has been done), please be aware that this command must be used when you change modules. You will find that **LAYOUT** and **ROUTER** drawings **LOOK** similar on the screen, but are quite different in the internal structure. Be careful.

The Auto-Board System II also makes use of the **SET** command for leaving a message for the software. As you learned earlier, the **PATH** command is only useful for finding program files. The netlist reader in the auto-router uses a database to recognize the configuration of various parts that are used to layout a board. Since data must be in the current directory to be accessed by a program, it would be logical to assume that the database files would have to be copied into each working directory to be used. Actually, there is a way to get to data outside of the working directory. But to do this, the program has to know exactly where to look. As in the case of the **ACAD** entry which allows **AutoCAD** to find secondary data files, the **SOFTWEST** entry allows the netlist generator and the netlist reader to find the location of their databases. You will need to set this message to the directory with the database files. If you followed the installation procedure, the database files are in the **SOFTWEST** directory. The message that you will leave will be entered (using the **SET** command) as:

```
set softwest=\softwest
```

Since the ACADCFG and SOFTWEST entries will not be changing for most of you, it might be a good idea to put these commands into the AUTOEXEC.BAT file. This file is a sequence of MS-DOS commands that you wish to have executed when the computer is first turned on (or re-started with the ALT-CTRL-DEL combination). The AUTOEXEC.BAT file must be in the root directory of the hard disk to be effective. If you don't have an AUTOEXEC.BAT file, it comes very highly recommended. See an MS-DOS or PC-DOS reference guide.

Chapter 3

Schematic Capture

3.1 Introduction

This chapter details the working mechanisms of the schematic capture portion of **The Auto-Board System II**. This chapter is divided into three parts. These parts deal with the use of **AutoCAD** and the schematic capture library to create a schematic diagram; the use of the schematic capture software to convert the **AutoCAD** drawing files into a netlist; and a look at the batch file that is used to control the process.

3.1.1 Before You Start ...

Review the following procedure and take necessary action. This process is a one-time-only procedure.

When a schematic entry session starts with **AutoCAD**, you will be asked to select the size of drawing that you will be producing. Your choices will be "A" size through "D" size. The menu item, when it is selected, will initialize **AutoCAD** with the necessary layers, limits, snap, grid and other parameters for the successful entry of a schematic. One of the last things that the menu macro will do is to **INSERT** a block. The intent of this block is to represent the borders and signature block of a sheet of paper of the selected size. Understandably, the names of the blocks are:

- SHEETA.DWG
- SHEETB.DWG
- SHEETC.DWG
- SHEETD.DWG

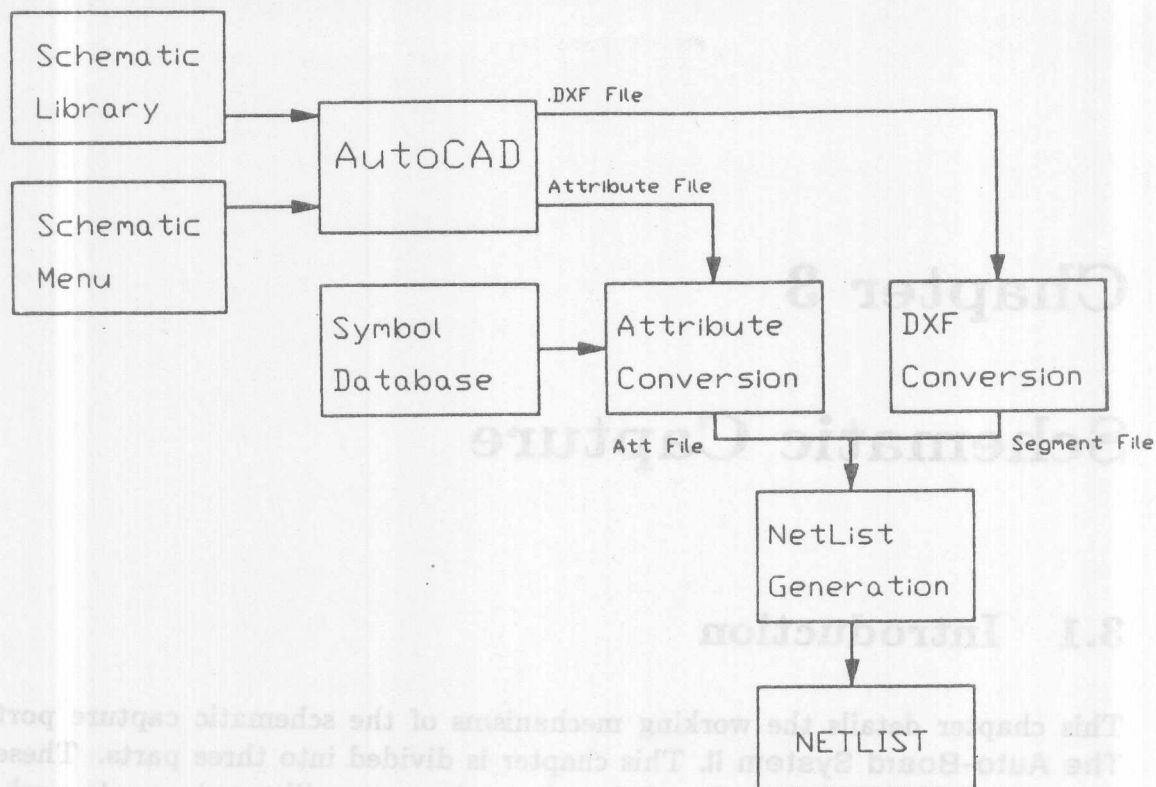


Figure 3.1: Organization of Schematic Capture Module

Since everyone has a different style of title block, we thought that it would be a bit presumptuous to force one upon you. But, then we know that some of you want to get started as quickly as possible. Here's what we've done. We've provided SHEETA.DWG and SHEETB.DWG in the schematic capture library (complete with script source). Use them, modify them or discard them.

However, for SHEETC.DWG and SHEETD.DWG, you are on your own. Use **AutoCAD** to design the border and signature and title block and save the results in the schematic capture library (SC). Use the script files for SHEETA and SHEETB as examples.

3.1.2 Overview

The majority of the work that you will be doing with the schematic capture module will actually be the use of **AutoCAD**. The **Auto-Board System II** software that was delivered to you operates on the output files of the **AutoCAD** drawing. Therefore, we will concentrate on the use of **AutoCAD** and drawing library. Everything else is automatic outside of the **AutoCAD** environment.

Figure 3.1 illustrates the organization of the schematic capture module. Notice that the only inputs to **The Auto-Board System II** software are **AutoCAD** files and

a parts database. You will not be expected to do anything interactively while the programs run.

3.2 Using Schematic Capture

This discussion about using the schematic capture module is divided into three parts (That sounds like something from Julius Caesar). These parts are:

- Starting the system with the batch stream
- Using AutoCAD to draw the schematic
- Finishing the session

3.2.1 Using the Batch File

To make life easier and the whole operation more successful, a batch file, AutoCAD menu and start-up script have been provided. These were loaded when the system was installed on your disk.

All that is necessary now to run the system is to select a name for the drawing and start the batch file.

Selecting a Schematic Name

You will be producing several drawings during the design cycle that you are starting with a schematic drawing. It is beneficial that you systematically name the drawing that you are producing so that you don't get confused. One of the by-products of confusing and similar names is that drawings can be lost when a drawing is saved and there is already another (completely different ... and quite valuable) drawing by the same name ... bye-bye drawing. This situation can also exist if the schematic drawing is the same name as the layout drawing of the board and the same name as the output of the router. It gets rather difficult to keep things straight.

Here is the solution that we've come up with:

- MS-DOS filenames can be a maximum of 8 characters long (FACT)
- Pick a PROJECT name of 6 or fewer characters
- Append "-S" to the PROJECT name for the schematic name

- Append “-L” to the PROJECT name for the layout name
- Append “-R” to the PROJECT name for the ratsnest name
- Use the PROJECT name for the routed drawing.

For instance, project “TEST” would be the base name. “TEST-S” would refer to the schematic related files. “TEST-L” would be the corresponding name for layout files.

This strategy is automatically implemented when the batch files of **The Auto-Board System II** are used. You will see that “TEST” is the only name that has to be given to any of the batch files in the system. You may see some of these names floating around and we want you to know what they mean.

Running Schematic Capture

The only things that you will have to know to start the schematic capture sequence are the names of the schematic capture batch file, the working disk directory and the project (the name of the circuit).

The name of the project is up to you. Following the instructions in the previous section, pick a name which has 6 or fewer characters. For illustrative purposes in this guide, we'll use the name “TEST” for our project name.

The name of the working directory is also up to you. In the installation chapter of this guide, you were shown how to create a working directory.

The name of the batch file which controls the schematic capture is SC. The batch file will initiate the **AutoCAD** program and the programs of **The Auto-Board System II** as they are required.

Before starting the batch file, make sure that the current directory on your disk is set to the working area that you should have created to hold all of the working drawings. Using PCB as the name of such a directory, enter:

```
cd \pcb
```

Start the batch file by entering:

```
sc test
```

Notice that the PROJECT name of the board is used to name the drawing (and subsequently the netlist).

A detailed discussion of the batch file will be presented in the last section of this chapter.

If things go wrong: If you should receive a *Bad command or file not found* message from MS-DOS, it means that the operating system was unable to locate the batch file or one of the programs mentioned in the batch file. There are several things that you should check:

1. Proper setting of the *PATH* command. The path should include the directories of **The Auto-Board System II** software (**SOFTWEST**) and **AutoCAD** (**ACAD**). Other directories may be in the list, but your path should contain (at least):

`PATH=\SOFTWEST;\ACAD`

Use the *SET* command to check the value of the path.

2. Spelling of directory names. **SOFTWEST** and **ACAD** are the primary ones here. If these aren't correct, rename the directory or copy its contents into a correctly named one.
3. Existence of the program. Check that the program is resident in the **SOFTWEST** or **ACAD** directories. The program name will have the file suffix of ".EXE".

3.2.2 Drawing the Schematic

If everything goes according to plan, the schematic capture batch file will set the proper secondary library for **AutoCAD** and load the **AutoCAD** program.

The batch file checks to see whether this is a new drawing. If it is, a script file is used to initialize the environment with a "NEW DRAWING" command for the **AutoCAD** command menu and the selection of a menu for the drawing editor.

Old drawings use a different script file to get into the **AutoCAD** drawing editor. The "OLD" script file has the required command of "2" for the command menu. Since the menu name is saved in an old drawing, there is no need to further initialize the drawing editor. The drawing editor automatically loads the schematic capture menu.

You will automatically arrive in the **AutoCAD** drawing editor. As **AutoCAD** is activated and works its way to the drawing editor, it will pass through and display the familiar main menu items and responses. You are not required to make any entries at this time. All responses and commands are being made by the script file that was activated by the batch stream.

The Schematic Capture Menu

The schematic capture menu (SC.MNU) is provided in an ASCII format. It is in the SC directory. It can be edited with a text editor or word processor. For instance, since the screen size varies from video board to video board, some entries may not fit on your screen. The menus have been built to fit on a 20 line menu area. All of the entries are there. They probably need moving up a little bit. Likewise, to make the menu somewhat more readable, you may want to expand the individual menu items with blank lines or **AutoCAD** editing commands. Some sub-menus title lines have a number following the name of the menu item (in the menu file, not one the screen). This number is the line number on which the menu item is to start. Change this number to move the sub-menu up and down on the screen. Feel free to do so. Save a copy of the original because you become responsible for the maintenance of the modifications. We don't fix modified menus.

You may want to insert the menu into a larger menu, such as the **AutoCAD** supplied menu or one with tablet commands in it. If this is the case, add some lines to the main menu entry point:

```
[SCHEMAT]$s=sapture  
[CAPTURE]$s=sapture
```

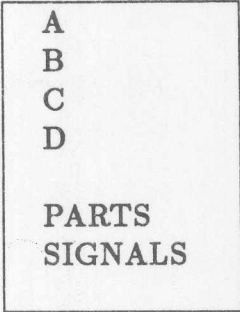
so that you can get to the added text.

And at the top of the schematic capture menu (that you will merge with the larger menu, add the line:

```
**sapture
```

Remember to add a line in the schematic capture menu so that you can get back to the main menu.

The first group of menu items that you will encounter is the drawing size and initialization menu. Your selections are:



A
B
C
D

PARTS
SIGNALS

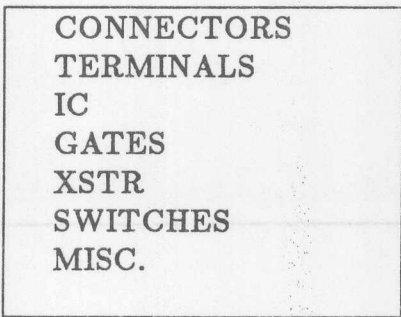
When starting a new drawing, you are required to set a drawing size. This size should correspond to the size of the paper that you are using on your plotter. Select a size A through D with your pointing device (cursor keys, tablet pointer or mouse). The macro will execute several **AutoCAD** commands to set up the schematic environment.

If you are producing MIL-SPEC drawings, you may find it convenient to select a "B" size drawing and plot it at 2:1 to produce a "D" size drawing with properly sized symbols.

The "PARTS" and "SIGNALS" entries are for reaching the interior menus for an old drawing.

Figure 3.2 shows a typical drawing area with an initialized screen.

With the completion of the initial set-up, another menu has appeared. This menu is for selecting the categories of symbolic circuit elements that are available to you. This is the Schematic Capture Parts menu. The categories of parts on this menu are:



CONNECTORS
TERMINALS
IC
GATES
XSTR
SWITCHES
MISC.

In general, these menu items refer to the following categories of schematic symbol:

CONNECTORS The connectors menu item allows the construction of connector block symbols for headers and edge connectors; and it also allows for the insertion of single male and female pins for scattered references to connectors and test points.

TERMINALS These symbols are not representations of physical components, but are used as a method of tagging a signal line or component lead with a signal name. All pins tagged with a terminal carrying the same name will be grouped together logically for routing.

IC This selection leads to a menu of generic symbol blocks representing ICs. The pins on these blocks are numbered in the manner of the physical IC. An attribute prompt will request the device specification so that you label the part with a specific part type (i.e., 74LS123). Custom IC libraries can be built following the instructions in chapters 6 and 8 or purchased separately.

GATES This selection leads to two menus of generic gate symbols. These symbols do not represent specific parts or technology (CMOS, TTL, etc.). To meet this generality, attribute prompts will request specific pin numbers whenever a symbol is inserted into the drawing. Custom gate libraries can be built following the steps in chapters 6 and 8 plus taking a look at the construction details of the script files provided with the system.

XSTR This menu item leads to a menu of transistor symbols. Mirror images of the symbols are included so that you can meet the design demands of push-pull configurations and other conventions.

SWITCHES Selecting this menu item brings up a menu with a set of standard switch symbols.

MISC. This menu covers a group of general schematic symbols of discrete parts. This group includes resistors, capacitors, LEDs, diodes and several others.

Drawings of the symbols in the generic library may be found in the appendix.

General Selection Procedure

Select and pick categories and sub-menus until you get to the part that you wish to place on the drawing. The following sequence is the GENERAL steps used to place a part:

1. Select part from menu
2. When prompted for x and y co-ordinates, move the **AutoCAD** cursor to the proper position. Press the "PICK" button or RETURN key on the keyboard.
3. When you are prompted for DEVICE NAME, enter the name of the part. This must be a unique reference identifier that will identify the part through ALL phases of The Auto-Board System II. The default value of the prompt should not be selected; it is there as a reminder of standard part prefixes.

4. When you are prompted for DEVICE TYPE, enter the specification of the part. This may be a part designator (74LS00) or a part value (2.7K ohms). This entry is optional. It will show up on the silk screen drawing of the routed board.
5. Several of the generic parts in the library may represent only a portion of a part. For instance, a 2 input NAND gate could be 1/4 of a 74LS00 package. In this case, the exact pins on the physical component will not be known by the NAND template. One gate may use pins 1,2 and 3, while another uses 4, 5 and 6. To clear up the ambiguity, you will be instructed to enter the exact pins that you wish to use. The prompts will be on the order of:

Pin for INPUT #1: 4 (*your response*)

Pin for INPUT #2: 5

Pin for OUTPUT: 6

Rotations and Reflections On several parts, there is a question of orientation. To eliminate the upside-down text for device specifications which can result when blocks are freely rotated in AutoCAD, rotations and mirroring of several parts (transistors, diodes, et.al.) have been implemented as separate drawings. When you select a specific symbol, you may be prompted with a positioning or rotation selection. When you choose one, you are selecting a drawing to insert. This is not the same as entering an angle for the insertion of a block in AutoCAD.

Some symbols selections will use the terms *TO THE LEFT* or *TO THE RIGHT*. This means that the main portion of the drawing will be to the left or right of the cursor position. The cursor position will be one (if not the only) of the points that connecting lines will be attached.

The gates in the GATES menu may be inserted in two ways. These two ways are prompted in the menu as:

IN==>OUT

OUT<==IN

These prompts refer to the side of the drawing that contains the input and output leads. For input leads on the left side and output to the right, select the first menu item. For right to left flow, select the second. Figure 3.3 shows the results of inserting a NAND2 (2 input NAND gate) with both orientations.

Up and down selections are not presented. Rotation of the block is permitted, but the text will rotate with it.

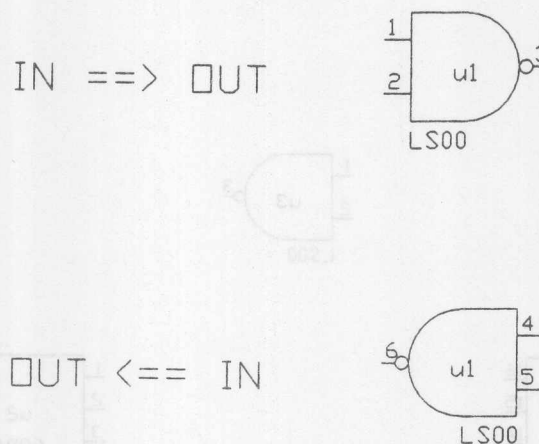


Figure 3.3: Orientation in Gate Symbols

Use the **AutoCAD MOVE** command to change the placement of symbols on the screen.

Figure 3.4 shows a couple of the symbols that have been selected from the menu and placed into the drawing area.

Connecting Everything Together

There are two ways in which you may show that components are connected in the schematic. The first method is with **TERMINALS**.

Terminals The terminals in the schematic library are not really components at all, but are handy methods of labeling pins on parts. Pins of the same label will be connected by the router. Several forms of terminals are available. They are:

- The "Ball" See Figure 3.5
- Offsheet Tags See Figure 3.6
- Bussing See Figure 3.7
- Ground See Figure 3.8

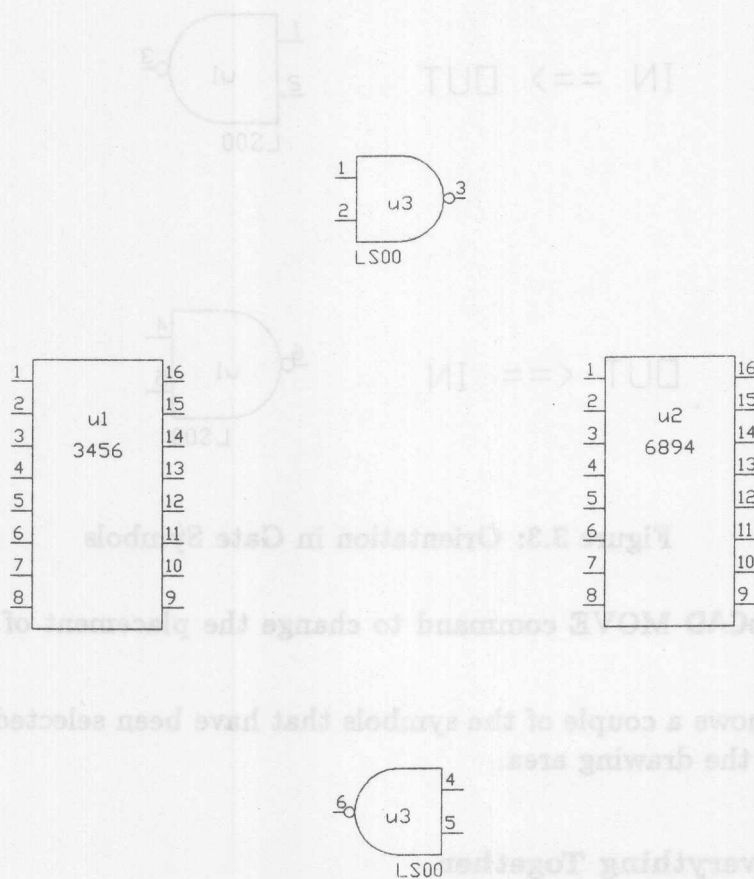


Figure 3.4: Schematic Symbols in the Workarea

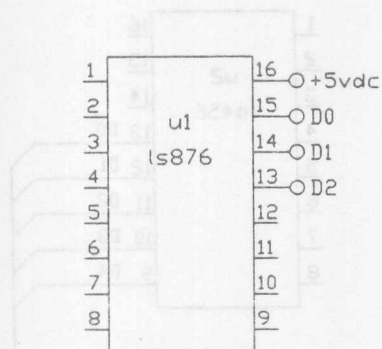


Figure 3.5: The "Ball"

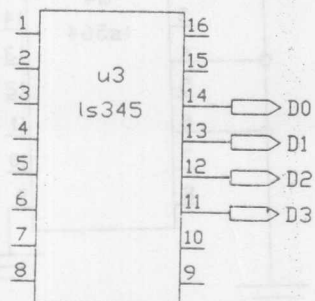


Figure 3.6: Offsheet Tags

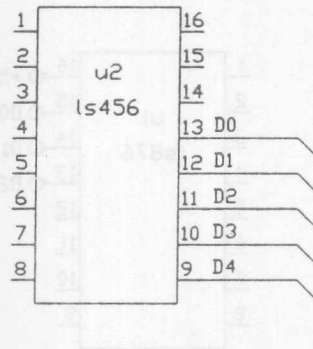


Figure 3.7: Bussing

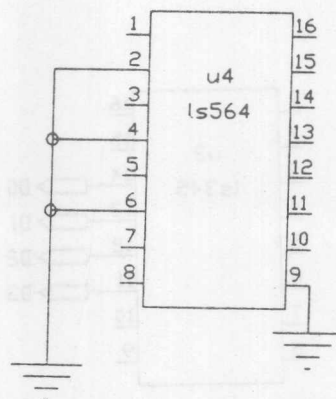


Figure 3.8: Ground

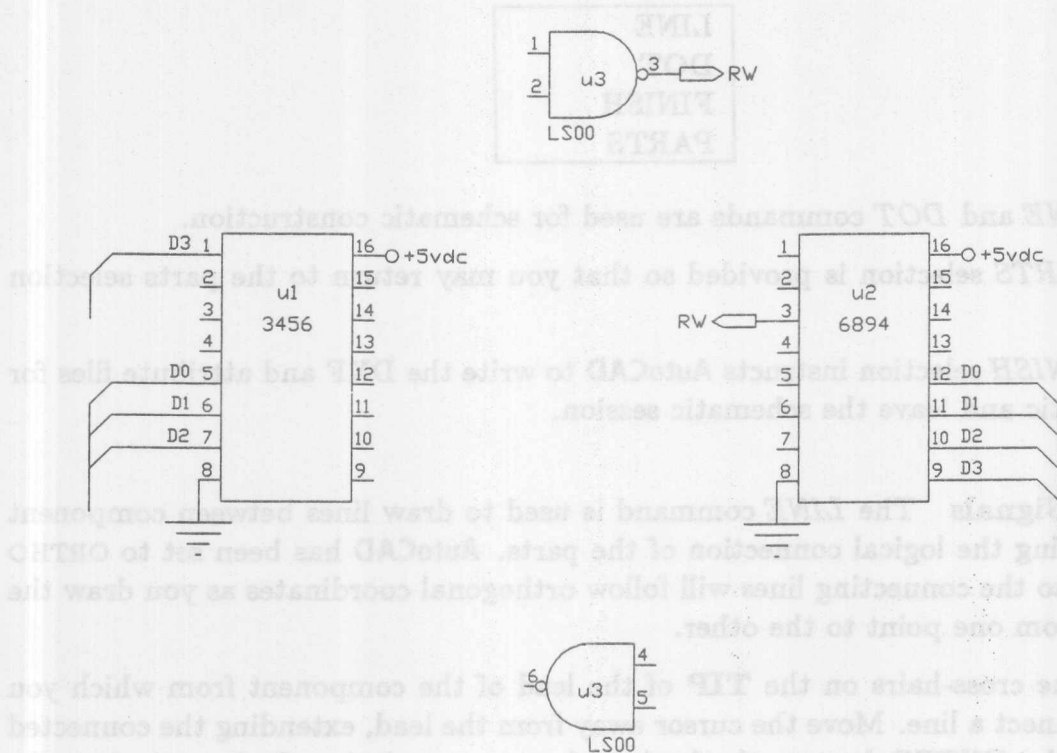


Figure 3.9: Signal Terminals in the Schematic

These represent the signal terminals available with The Auto-Board System II. Other forms of terminals can be easily added to the libraries.

The signals names may appear on several schematic drawings, linking the drawings together logically through the generated netlists.

Figure 3.9 shows the results of attaching several terminals to the previously inserted symbols.

Signal Lines This section covers the second method of connecting parts together. This method involves connecting points visually and logically with line segments using normal schematic drafting methods.

Referring back to the screen menu of your AutoCAD session, select the SIGNALS menu item. This selection performs some changes to the AutoCAD operating envi-

ronment (changes the layer) and presents a new menu.

Your selections on the SIGNALS menu are:



The *LINE* and *DOT* commands are used for schematic construction.

The *PARTS* selection is provided so that you may return to the parts selection menus.

The *FINISH* selection instructs AutoCAD to write the DXF and attribute files for the schematic and leave the schematic session.

Drawing Signals The *LINE* command is used to draw lines between component leads, showing the logical connection of the parts. AutoCAD has been set to ORTHO operation, so the connecting lines will follow orthogonal coordinates as you draw the segments from one point to the other.

Place the cross-hairs on the **TIP** of the lead of the component from which you want to connect a line. Move the cursor away from the lead, extending the connected line. Use the ENTER key on the keyboard or your pointing device to select the endpoint. Continue drawing line segments until you reach the **TIP** of the lead of the target component. Select that endpoint with the ENTER key. A second ENTER will be required to terminate the line.

Figure 3.10 shows the schematic being developed to the point of connecting several pins with signal lines.

Lines must TOUCH the end point of the lead (but not overlap)

One of the benefits of ORTHO is that once the first endpoint of the line is established, the line will extend to the cross-hair of the dominant orthogonal axis. This allows you to move the unused cursor cross-hair away from the extending line, thereby increasing visibility for you. It is not necessary to have the cross-hairs "zeroed-in" on the selected coordinate.

Crossing Lines This session is not a routing exercise so you do not have to be concerned with crossing lines. It is perfectly okay to cross lines. Crossing lines do not designate a connection. However, lines meeting at a common point DO constitute a connection. It is important that you recognize this fact since it is virtually impossible

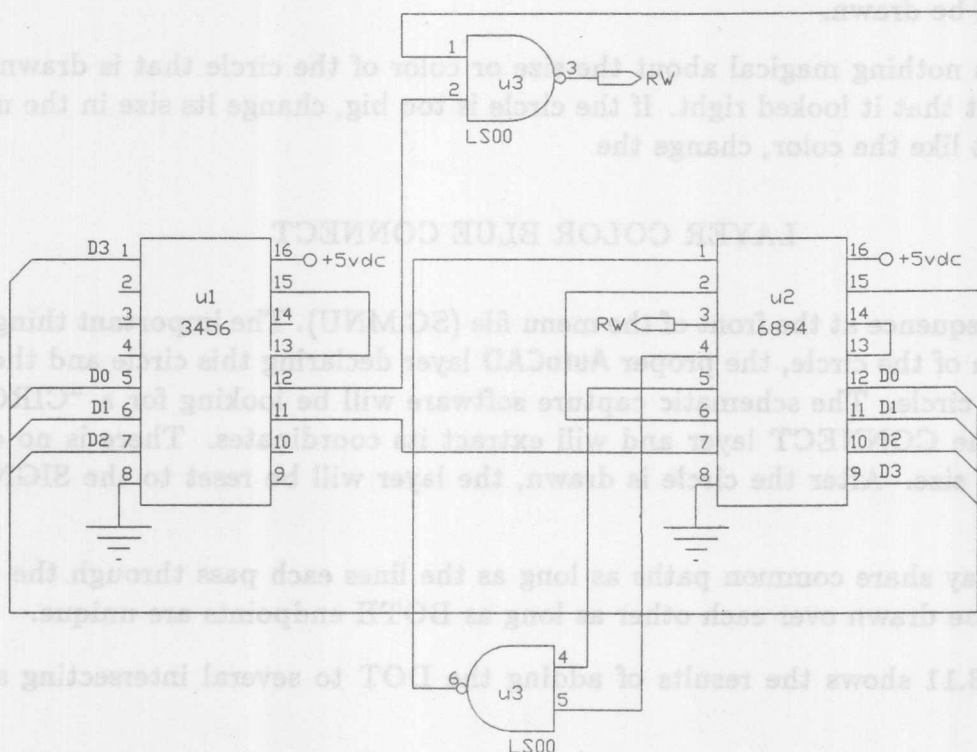


Figure 3.10: Connecting Parts with Signal Lines

to tell the difference between four lines that terminate at a common point and two crossing lines. Not realizing that there is a connection implied by "crossing lines" can cause erroneous netlists to be generated. Use the *DOT* command to denote these kinds of connections.

To connect crossing lines, T-joints and ambiguous points (corners are recognized, no problem), select *DOT* from the menu. This will change the layer (required by the capture software) and prompt for a location. HIT the crossing point dead-center and a circle will be drawn.

There is nothing magical about the size or color of the circle that is drawn. We just thought that it looked right. If the circle is too big, change its size in the menu. If you don't like the color, change the

LAYER COLOR BLUE CONNECT

command sequence at the front of the menu file (SC.MNU). The important things are the location of the circle, the proper **AutoCAD** layer declaring this circle and the fact that it is a circle. The schematic capture software will be looking for a "CIRCLE" entity on the CONNECT layer and will extract its coordinates. There is no check for color or size. After the circle is drawn, the layer will be reset to the SIGNALS layer.

Lines may share common paths as long as the lines each pass through the path. Lines may be drawn over each other as long as BOTH endpoints are unique.

Figure 3.11 shows the results of adding the DOT to several intersecting signal lines.

3.2.3 Special Techniques

Since **AutoCAD** is not a dedicated printed circuit design system, some special methods are used to perform functions that might be considered standard equipment in a pcb system. The following discussions detail the creation of memory arrays in the schematic and the methods of defining the power and ground pins for symbols that do not normally show them (in particular, gates).

Memory Arrays

The obvious reason for constructing a schematic drawing with **The Auto-Board System II** is to obtain a netlist for use in the auto-router. The primary means of communication with the router being the netlist.

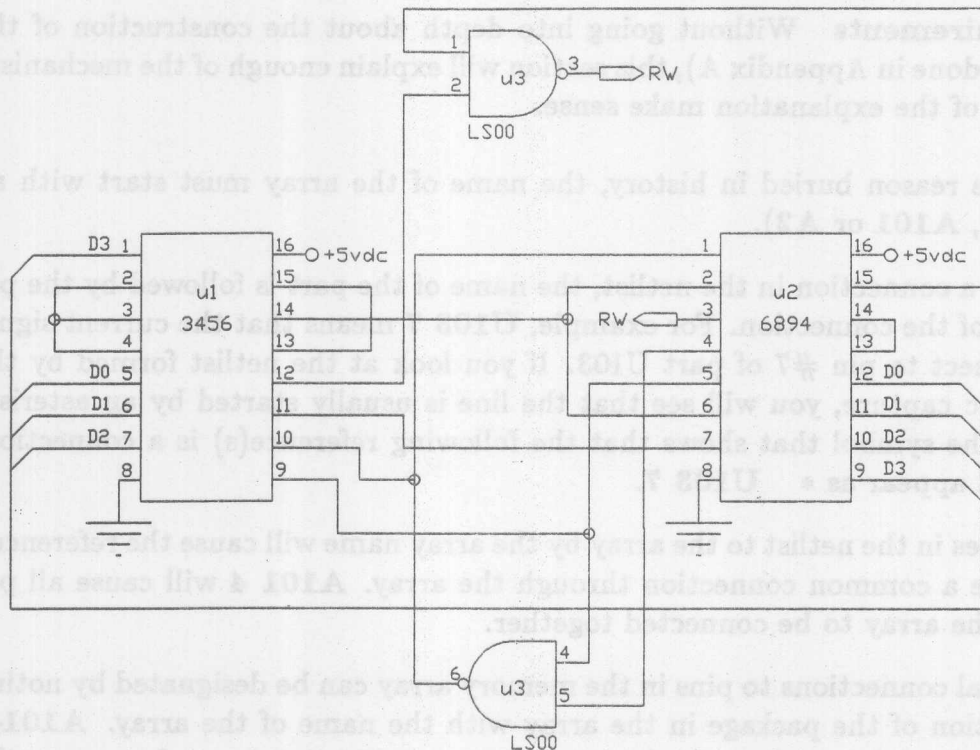


Figure 3.11: Crossing Signals Being Connected in the Schematic

One of the special operations of the router is the memory array processor. This program links the common connected pins in an array together with a set of woven tracks constructed for each instance of an array.

Notations in the netlist let this happen. The purpose of this section is to describe the schematic technique for producing the correct netlist entries for the proper operation of the array processor.

Netlist Requirements Without going into depth about the construction of the netlist (that is done in Appendix A), this section will explain enough of the mechanism to let the rest of the explanation make sense.

- For some reason buried in history, the name of the array must start with an "A" (i.e., A101 or A2).
- To show a connection in the netlist, the name of the part is followed by the pin number of the connection. For example, U103 7 means that the current signal will connect to pin #7 of part U103. If you look at the netlist formed by the schematic capture, you will see that the line is usually started by an asterisk. This is the symbol that shows that the following reference(s) is a connection. This will appear as * U103 7.
- References in the netlist to the array by the array name will cause the referenced pin to be a common connection through the array. A101 4 will cause all pin #4s in the array to be connected together.
- Individual connections to pins in the memory array can be designated by noting the position of the package in the array with the name of the array. A101-3 refers to the third package in the array, A101. A101-3 5 is a reference to the fifth pin of the third package in the array, A101. This may be a *chip select* line or other signal that must be routed to a unique position in the array.

That covers the details of the netlist enough to let you understand the construction of the schematic for an array.

Showing Common Connections From the construction details of the netlist, it should be clear that the difference in a signal being routed to all packages or to a single package is the form of the array name. Either the package designator is there ("A101-3") and the signal is to a single pin; or it isn't there ("A101"). This section will cover the common connections.

- Select an IC from the menu or custom library for the memory package that you will be using.

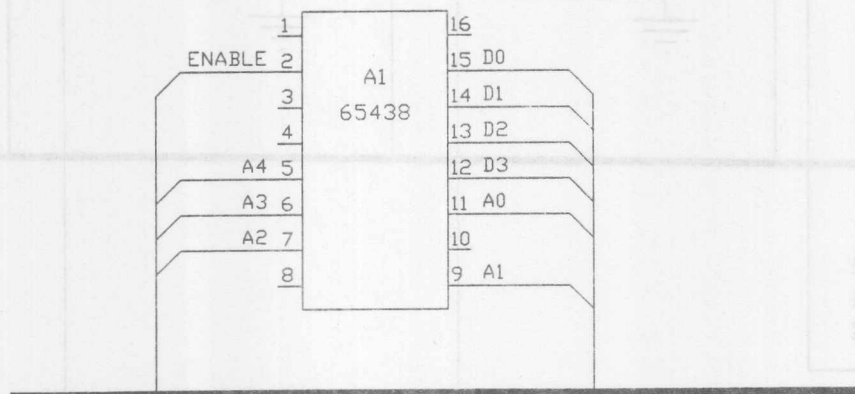


Figure 3.12: Array - Common Connections

- Position it on the screen in a convenient place.
- When you are requested for a name, give the name of the array. Remember to start the name with an "A". For example, use **A1**.
- Make signal connections to this symbol as you normally would.

Any signals connected to this symbol will now represent common connections.

When the netlist is generated, the name of the array, **A1**, will be mated with the pins that show a connection. The netlist generator will combine these to form the proper netlist notation (i.e., **A1 5**).

Figure 3.12 shows a memory symbol with common connections. The buss terminals are used to indicate the origin of the signal being from the signal buss.

Showing Unique Connections As you have seen, the difference between the common connections and the unique connections is in the name of the package. To show a unique connection, use a symbol with the name of an individual package.

For a four package array (not enough room to show a 9 package array):

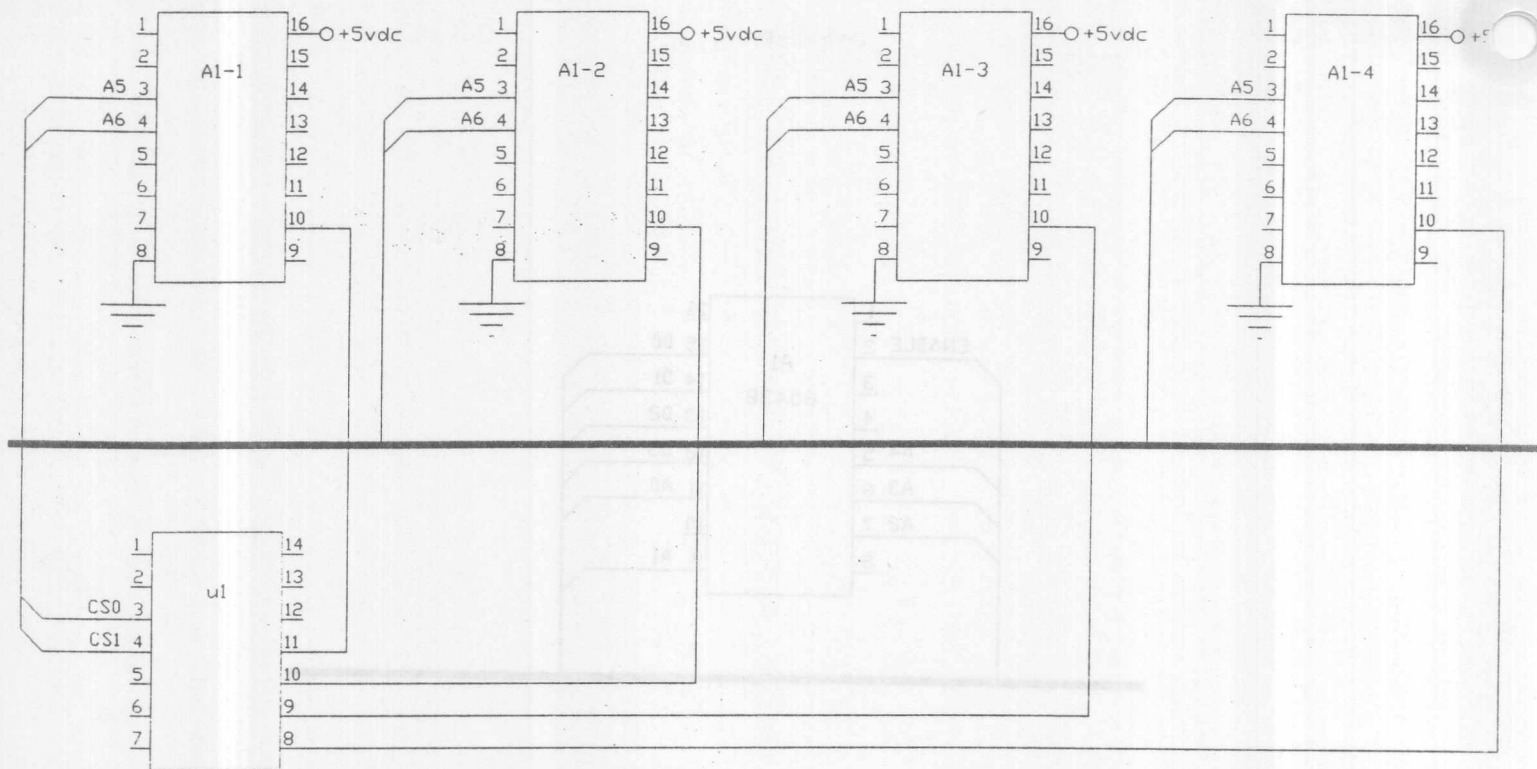


Figure 3.13: Array - Individual Connections

- Select and place four IC symbols from the menu or a custom library.

- Give the symbols names that show their respective positions:

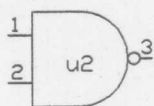
- A1-1
- A1-2
- A1-3
- A1-4

- Make signal connections to these symbols as you normally would.

Any signals connected to a symbol will now represent a unique connection. The router will run a separate trace to that pin.

When the netlist is generated, the name of the array package, **A1-3**, will be mated with the pins that show a connection. The netlist generator will combine these to form the proper netlist notation (i.e., **A1-3 12**).

Figure 3.13 shows a series of memory symbols with unique connections.



No obvious Power and Ground

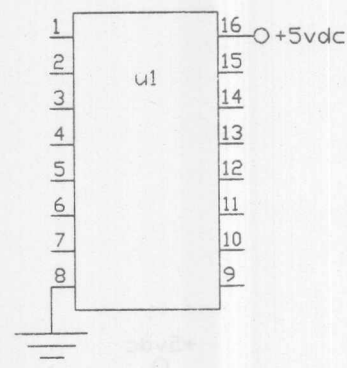


Figure 3.14: Power and Ground Dilemma

Power and Ground

One of the problems with using generic schematic symbols (We encourage the use of custom libraries.) is the lack of those extra pins for defining power and ground connections for the package. These pins are certainly available for the generic ICs, but are absent from the gates.

One alternative is to provide an extra set of symbols for each logic group that would contain an extra pin for power and ground. If you are interested in large symbol counts, this would certainly build up the number of parts in the library.

The method that we chose was the introduction of one additional symbol to the library. The symbol is a power and ground pin symbol. Figure 3.15 shows the symbol that The Auto-Board System II uses to document the power and ground pins for a device.

The use of this symbol is similar to other generic symbols in the library.

- Select the PWR/GND item in the IC menu.
- Position the part on the drawing. We suggest that you put all of the power

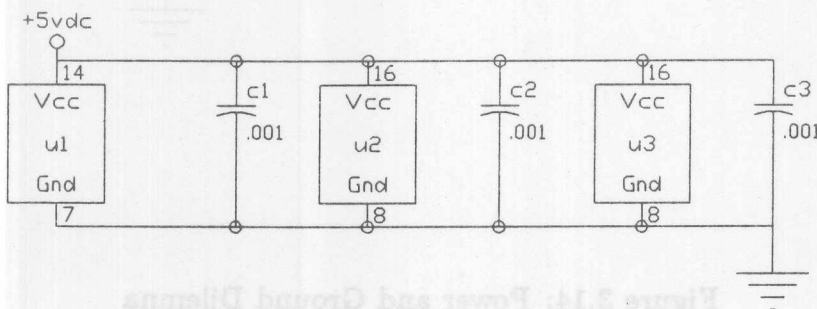


Figure 3.14: Power and Ground Dilemma

Power and Ground

One of the problems with using generic schematic symbols (We encourage the use of custom libraries) is the lack of those extra pins for defining power and ground connections for the package. These pins are certainly available for the generic ICs, but are absent from the gates.

One alternative is to provide an extra set of symbols for each logic group that would certainly build up the number of parts in the library.

The method that we chose was the introduction of one additional symbol to the library. The symbol is a power and ground pin symbol. Figure 3.15 shows the symbol that The Auto-Boast System II uses to document the power and ground pins for a device.

The use of this symbol is similar to other generic symbols in the library.

• Select the PWR/GND item in the IC menu.

• Position the part on the drawing. We suggest that you put all of the power

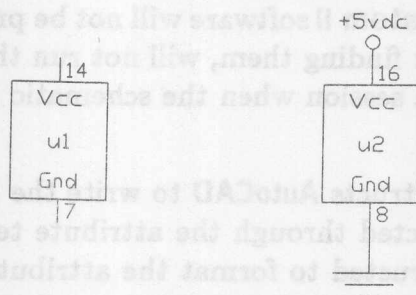


Figure 3.16: Power and Ground Symbol

and ground symbols in a line towards the bottom of your drawing where they will be out of the way. Figure 3.15 shows just such an arrangement. Notice how the bypass capacitors can be put into the drawing along with the power and ground symbol, keeping them together without cluttering the drawing with extra symbols.

- When prompted for the name of the device, enter the reference designator of the device for which this symbol is to be associated. There will not be a prompt for the type of the device.
- The next prompt will be for the number of the *POWER* pin. Enter the number.
- The last prompt will be for the ground pin. Enter it.

Even though the power and ground pins have been designated, they need to have a signal attached to them so that the netlist generator can output a net for this device. Use the ball and ground symbols to attach the proper signals. Figure 3.16 shows the bare-bones symbol with and without power and ground attached.

Using FINISH

All AutoCAD commands can be used to edit the drawing until it is as you think it should be. Select the *FINISH* command on the SIGNALS menu to leave the session. This command will issue an *END* command so that the drawing will be saved for later sessions.

You may choose to use the AutoCAD *END* command to save your drawing instead of using the *FINISH* command in the menu. If you do this, the output files that are used by The Auto-Board System II software will not be produced. The batch stream checks for these files, and not finding them, will not run the software. This would be a typical method of ending a session when the schematic is not yet ready for netlist generation.

The *FINISH* selection instructs AutoCAD to write the DXF and attribute files to the disk. Attributes are selected through the attribute template file, SC.TXT. This template file has been constructed to format the attributes in the manner required by the netlist generation software. It is possible to "overflow" some fields in this template. This can be done if you select a device name or specification that is longer than the size stated in the template. You may change template SIZE values for this reason. However, we will not tolerate any other modifications of this file. You might as well kiss technical support, "Good-bye" if we find that this file has been modified. Is that strong enough?

3.3 Schematic Capture Software

The following section describes the functions of the software programs and batch streams that generate the netlists from your schematic drawings.

The schematic capture software is responsible for interpreting the DXF and TXT files that AutoCAD writes. These files contain information about the nature and content of the AutoCAD drawing. This information is extracted and rearranged to produce a netlist of the circuit that you designed. This task is accomplished through three programs:

ATTCNV Attribute File Conversion

DXFCNV DXF File Conversion

NETGEN Netlist Generation

Their use is extremely simple but their execution should be controlled by a batch file to save you some typing and "sitting around" time. The batch file, SC.BAT, has

been provided for you for this control function.

3.3.1 Attribute Conversion

The ATTCNV program reads the attribute file (".TXT") written by AutoCAD when you select *FINISH* on the menu. With positioning information supplied by the attribute file and a special database of pin offsets, ATTCNV writes a file of component lead coordinates found in the drawing. This program could be activated by the following command line:

```
attcnv schematic-name
```

3.3.2 DXF Conversion

The DXFCNV program reads the drawing exchange file (".DXF") written by AutoCAD when you select *FINISH* on the menu. This program extracts the network of connections from the schematic drawing. This program is activated by the following command line:

```
dxfcnv schematic-name
```

3.3.3 Netlist Generation

The NETGEN program combines the information from ATTCNV and DXFCNV to make connectivity relations. Knowing the connectivity of the schematic, a netlist can be produced. This program will also issue warning messages for leads that have no connections or signal names that share a common connection. This program is activated by the following command line:

```
netgen schematic-name
```

This program must be run after ATTCNV and DXFCNV have completed.

3.4 The Batch File Examined

Operation of the schematic capture software is not difficult but life is so much easier when there is an automatic means of doing the work. Work smart, not hard. The Schematic Capture (SC.BAT) batch file is provided to perform the sequence of operations of schematic capture. The batch file covers the following operations:

1. Selecting the schematic symbol library for **AutoCAD**. This is done with the "SET ACAD=" command. In the batch file, this is seen as:

```
echo off
set acad= \sc
```

2. Starting **AutoCAD** with initialization for a new drawing. Facilities of MS-DOS allow a batch file to check for the existence of a file. Depending upon whether a file exists or not, it can allow certain logical decisions to be made. In the case of this batch file, the command to start **AutoCAD** is tailored to include a script file to request a new drawing and to bring in the schematic capture menu. You can edit the script file to do other things if you wish. Old drawings don't require such initializations but the command line does include the script file reference to keep things consistent.

The batch file does this with:

```
if exist %1-s.dwg goto old
acad %1-s sc
goto join
:old
acad %1-s old
:join
```

3. Starting the schematic capture software. Relying upon the check for existing files, the batch file runs the ATTCNV, DXFCNV and NETGEN programs. They are triggered by the existence of certain files written by **AutoCAD**. If these files are not found (meaning that the schematic is probably not done yet), the execution of the programs is by-passed.

The batch uses the following lines to perform this:

```
if exist %1-s.txt attcnv %1-s
if exist %1-s.dxf dxfcnv %1-s
if exist %1-s.att netgen %1-s
```

4. Cleaning up the intermediate files from the session.

```
if exist %1-s.dxf del %1-s.dxf
if exist %1-s.txt del %1-s.txt
if exist %1-s.att del %1-s.att
if exist %1-s.seg del %1-s.seg
```

The "if exist" passages aren't really necessary since MS-DOS can hardly delete a file that doesn't exist. However, we have found that the user community feels so much better when the message *File not found* does not appear on the screen.

There are no conventions for naming schematic drawings. However, the batch file that is delivered to you does promote the use of the "-S" suffix for schematic drawings. When the batch file is started by entering:

```
sc test
```

the internal commands append the "-S" to the name of the project, "TEST". You may remove the suffix from the batch file commands if you wish to adopt a different convention.

While we are on the subject of batch streams, one other batch files is included on your distribution disk. It is on the first schematic capture disk and is called "PARTS.BAT". This stream should be started with SC (the schematic capture library) directory as the current directory. This batch stream will cause all schematic library drawings to be erased and re-generated from their respective script files. Use this batch stream whenever you have modified a part in the library and are not sure about replacing just one member. This is the batch stream that we use to generate the library from time to time.

The "if exist" passages aren't really necessary since MS-DOS can hardly delete a file that doesn't exist. However, we have found that the user community feels so much better when the message file not found does not appear on the screen.

There are no conventions for naming schematic drawings. However, the batch file that is delivered to you does promote the use of the "-S" suffix for schematic drawings. When the batch file is started by entering:

uc test

the internal commands append the "-S" to the name of the project, "TEST". You may remove the suffix from the batch file commands if you wish to adopt a different convention.

While we are on the subject of batch streams, one other batch file is included on your distribution disk. It is on the first schematic capture disk and is called "PARTS.BAT". This stream should be started with SC (the schematic capture library) directory as the current directory. This batch stream will cause all schematic library drawings to be erased and re-generated from their respective script files. Use this batch stream whenever you have modified a part in the library and are not sure about replacing just one member. This is the batch stream that we use to generate the library from time to time.

Chapter 4

Component Placement

4.1 Introduction

The Component Placement package (Layout) of The Auto-Board System II is a group of AutoCAD aids and software tools to assist you in laying out the physical components on a printed circuit board. The premise of the package is that you will be able to pick parts from a menu, place them onto the working drawing as they should appear in the routed board and automatically form the necessary files that are used by the auto-routing package of The Auto-Board System II.

Component Placement has four major parts:

- AutoCAD menu for picking and placing parts
- AutoCAD drawing library of part outlines
- The LAYOUT program
- AutoCAD script file with source code to part library drawings. This script is supplemented by several individual part outlines.

All human interfaces with this portion of The Auto-Board System II are handled through the AutoCAD program. Therefore, besides the drawing that you have at your disposal for laying out the board, you are able to use all of the commands that are normally available through AutoCAD (i.e., ERASE, ZOOM, MOVE, etc.).

4.2 Laying Out a Board

Once again, all of the work that you will have to do with the computer will be through **AutoCAD**. This will put you into a familiar situation with a minimum of new learning.

4.2.1 Starting the System

You will be activating **AutoCAD** and **The Auto-Board System II** software through a batch file command, *BOARD*. The name that you give to the command will be the same name that you used for schematic capture and will use in the auto-router. Refer to the section in the previous chapter about selecting a name for the project and how the batch files modify that name to keep all of the drawings separated. In the *BOARD* batch file, the "-L" suffix will be added to your project name.

Start the batch stream for laying out a board with the command line:

```
board test
```

You will recognize the name, *test*, as the sample name that we have been using throughout the examples. You will need to substitute the name of your project.

4.2.2 Setting Up Component Placement

The batch file will make some appropriate setting for **AutoCAD** so that it will be able to find the **LAYOUT** library.

AutoCAD will be the first program to run. The batch file will decide whether you are working on a new or old drawing and will activate **AutoCAD** in the appropriate manner. In either case, old or new drawing, the steps that are needed to get you to the drawing editor are handled automatically.

AutoCAD Initialization

When the *LAYOUT* menu appears on the screen, you will see several items to be selected. To set up the system for laying out a PC board, move the screen cursor to and select the *BOOKWORK* item. This will cause the menu to change to a series of commands for initializing the drawing session and for ending the session.

Select the *SETUP* menu item. With this selection, there is nothing that you will have to do (except wait for the completion of all of the imbedded commands). This

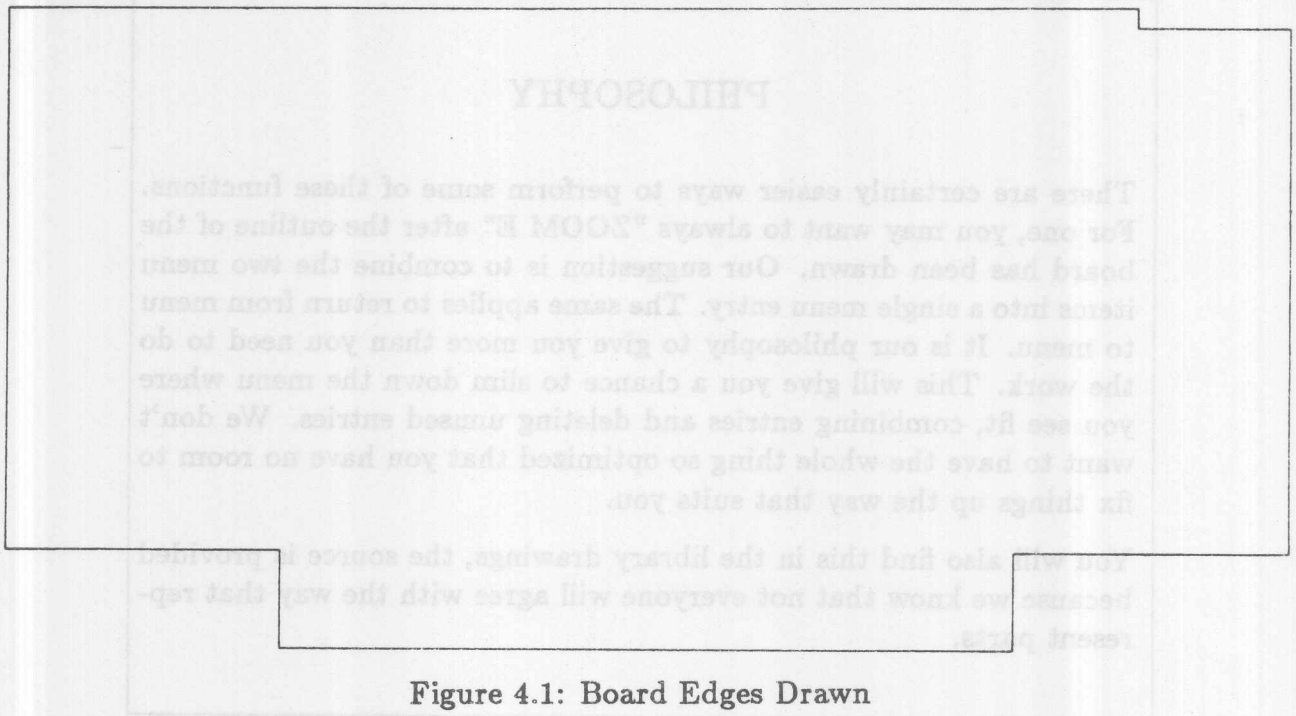


Figure 4.1: Board Edges Drawn

menu item sets the scene for board design. The layers are created, grid initiated and snap set.

Board Outline

When the *SETUP* function is complete, it brings you into the *EDGE* definition menu. You may wish to take advantage of the menu item that allows you to define the edges of the board. It may be your practice to layout out the board and then define a board boundary that fits the layout, (if you do, you are one designer in a million), but board outline layout now and processing later will proceed smoother if you do this now.

Select the menu item marked *EDGES* (not *EDGE OUT ... yet*). This menu item will perform several functions, but will finally result with a *LINE* command and a request for the starting coordinates of that line. Move the cursor to a comfortable starting location, marking each of the corners of the board (in order). Figure 4.1 shows the edges of a board drawn with the *LAYOUT* selections. Close the board outline with a finishing line or enter the letter "C" for *CLOSE* from the keyboard to close the outline. There is a *SIZE UP* entry that will perform a "ZOOM Extents" so that the board fills the screen. The *LAYOUT* selection returns you to the *BOOKWORK* menu and back to the main menu with the selection of another *LAYOUT*.

PHILOSOPHY

There are certainly easier ways to perform some of these functions. For one, you may want to always "ZOOM E" after the outline of the board has been drawn. Our suggestion is to combine the two menu items into a single menu entry. The same applies to return from menu to menu. It is our philosophy to give you more than you need to do the work. This will give you a chance to slim down the menu where you see fit, combining entries and deleting unused entries. We don't want to have the whole thing so optimized that you have no room to fix things up the way that suits you.

You will also find this in the library drawings, the source is provided because we know that not everyone will agree with the way that represent parts.

Hot Tip There is a selection of the *BOOKWORK* menu named *EDGE OUT*. This command will generate an AutoCAD DXF file so that the component placement program can figure out where the edges of the board are. The DXF file contains a huge amount of other stuff that isn't necessary to the production of parts definitions for the router. Therefore, the DXF file will be shorter and the component placement program will run faster if the DXF file is created immediately after the edges are drawn and before any parts are positioned. This explains the separation of the *EDGE OUT* item and the *FINISH* item which reports the positions of the parts. And it also explains why the menu returns to the *BOOKWORK* menu before returning to the main menu.

However, there are other features of the component placement phase that use the DXF file for data. Don't perform the *EDGE OUT* item until all of these items are in place or else they will not show up in the final drawing. These items are:

- Buss declarations
- Masked areas
- Board edges

If the only thing that you have on the board from this list is the board edge definition, then it makes sense to write the DXF file before the board becomes too cluttered.

If you are modifying an existing drawing, remember to perform the *EDGE OUT*. The old DXF file have be deleted by this time and the edges will have to be re-established.

4.2.3 Pick and Place Parts

The various menu items on the main LAYOUT menu lead to specific parts of the given category. Since the menu is longer than some of the screens currently in use, the menu has been split to allow a generous number of categories. Select the *MORE* item to see the second set of categories.

Device Select:	Device Select:
DIP	VIAS
ARRAY	DB-9/25
SIP	MASK
XSTR	EDGE
RESISTOR	BUSS
CAPACITR	OTHER
DIODE	-LAYOUT-
CONNECT	
-MORE-	

Another general word about the menu's construction: We think that we have removed all of the menu traps (that ability to get into a menu and no way out). These were usually caused by using video screens with fewer visible lines than there were menu items. But if you are unfortunate enough to get trapped, the title of the menu at the top of the screen is a menu return selection. Use it to get out of a dead end menu.

Available Parts

You will find in the menus that the integrated circuits and SIP are classified by pin count. The parts provided with the system are:

IC8
IC14
IC16
IC18
IC20
IC22
IC24
IC28
IC40
IC64
4 PIN
6 PIN
8 PIN
10 PIN
11 PIN

Resistors are classified by wattage (which determines lead spacing):

1/8 Watt	400 mil spacing
1/4 Watt	500 mil spacing
1/2 Watt	750 mil spacing
1 Watt	1000 mil spacing
2 Watt	1200 mil spacing

Capacitors are classified by lead spacing. There seem to be so many configurations of capacitors that this method of classification became the only suitable approach. There is nothing stopping you from using these conveniently spaced pads for other parts. For instance, the 100 mil spacing of the smallest capacitor is just right for LEDs, and some of the medium size values can be used for crystal mounting.

100 mils
200 mils
250 mils
300 mils
500 mils
600 mils
800 mils
1000 mils
1500 mils
2000 mils
2200 mils
2500 mils

Transistors are classified by can-type. Although the lead spacing on several of

the popular sizes is similar, variations have been introduced so that you can get the lead alignment that you require. For instance, TO202T (triangular mounting) and TO220T are basically the same package. However, the center lead on the TO202T is to the right of the others and TO220T is to the left. The silk screen pattern for both is the same; the body of the transistor is horizontal for mounting to a heat sink.

TO5
TO8
TO18
TO39
TO92
TO98
TO202T
TO202S
TO218
TO220T
TO220S
TO226

Details of Selection

To place one of these parts on the screen, select (pick) the part from the menu. Place parts on the board in the same manner as you did in the schematic capture:

- Pick the component that you wish to place from the menu
- Wait for the **AutoCAD** prompt for insertion point and rotation
- Enter the device name and optionally the device type

The device type entry in the schematic capture will override this entry, so most users leave it blank.

Position of Part **AutoCAD** will request that you designate a position on the drawing area for placing the part. If you're being serious about this, you will select a position which is entirely within the boundaries of the board. The router will do nasty things to circuit elements outside of the board boundaries (and it doesn't always ignore them).

Move the cursor to the position for PIN #1 of the device and press the appropriate selection button / key to mark the place.

Rotation of Parts AutoCAD will next ask for a rotation value. The parts in the library are aligned with PIN #1 at the TOP of the screen (resistors, diodes and capacitors are drawn vertically). On integrated circuits, the notch is the reference point. If the default rotation is what you want, press the selection button/ key. If you want the part rotated to its side or upside down, there will be a rotation sub-menu at the top of the menu area for selecting the orientation of the part at the appropriate time.

Rotation entry menus are provided on the menu screen. Rotation for ICs is by the location of the positioning notch. Rotation of discrete devices is by the position of pin #1 in relation to the rest of the part. The default rotation is 0 degrees. This value will be used with the simple entry of a carriage return. However, the menu item will not reset if you use a keyboard or digitizer entry. Use the menu escape procedure of selecting the title of the menu to get back to the regular menu items.

Enter Reference Designator AutoCAD now knows what, how and where to draw the part outline, but AutoCAD still has some questions to ask. All of the parts have attributes attached to them. These attributes are pieces of information that are carried with each individual part. Some of the attributes are preset to some constant value (not displayed) while other attributes are unique to each instance of the part and must be supplied by the user.

These user supplied attributes are the part name and specification. The first request will be for the name of the part. A default value is given. **Don't use it.** It is only a reminder of the prefix the name of the part. The name of the part must be unique. This is your responsibility. This attribute **MUST** be used.

The second request will be for a part value. In the case of resistors, the attribute will prompt with "OHMS:". This is a free form value for part value or internal part number for the later compilation of a parts list.

When these values have been entered, the part will be drawn on the screen with the attribute information displayed.

Continue picking and placing parts. You will realize the greatest efficiency if you pick and place all of your parts from one family at a time (all of the ICs and then all of the resistors, etc.).

4.2.4 Special Techniques

There are several types of operations that you can perform in the component placement session that will generate special parts. These parts include:

- Memory Arrays
- Edge Connectors
- Headers
- Signal Busses

Each operation requires the use of several levels of menus so that you can perform the entry sequences correctly.

Memory Arrays

Arrays of dual inline packages can be constructed for use as memory arrays in the router. These arrays will be handled by a special process in the router session. This process generates the “wavy” traces that connect the common pins in the array.

The *ARRAY* menu item in the component placement menu will help generate the proper declaration of a memory array so that the router will process it correctly.

Select the *ARRAY* item. The main menu will be replaced with a menu asking for array orientation. This may seem like it's getting the cart in front of the horse, but it actually causes menus with the rotation factors built in to them to be displayed. The rotation menu looks like this:

```
ARRAY-A
Select:
NOTCH
UP
LEFT
DOWN
RIGHT
-LAYOUT-
```

Select the orientation of the array the position of the notch. The menu will be replaced by a menu requesting the type of package that will be used. Regardless of the orientation, all of the resulting menus will look the same (with exception of the title at the top of the menu).

UP
ARRAY-A

IC16
IC18
IC20
IC22
IC24
IC28

-LAYOUT-

Select the type of package that you will be using. Locate the position of pin #1 of the first package in the array. Scaling and rotation values are already in place in the menu.

AutoCAD will prompt for the name of the array. There is a strange convention that is used at this time. Name the array with the prefix of "A", (i.e., "A1" or "A101").

The type of the component will also be requested. A response to this prompt is optional.

The **AutoCAD array** instruction is then started. All of the normal prompts are covered by the menu item. This is why the orientation was requested earlier. There are two entries that you will have to make.

- You will have to enter the number of units in the array. Depending upon orientation, you will be prompted for number of rows or number of columns. When the array instruction finally stops at one of these prompts, enter the number of packages in the array.
- You will have to enter the spacing between the units. In the normal use of the array command, the spacing is relative to the insertion point. However, in memory array design, the spacing that matters is the distance between packages. To aid in the conversion of unit spacing, the following menu is displayed. Please use it. It has been corrected for rotation and size of package.

Unit Spacing
100 mils
150 mils
200 mils
250 mils
300 mils
400 mils
500 mils

With the parameters to the array instruction in place, AutoCAD will step and repeat the package to build the memory array.

Edge Connectors

Edge connectors of several sizes can be constructed during this session of component placement. The Auto-Board System supports edge connectors of three spacings:

- .1 inch

- .125 inch

- .156 inch

The router has a special program for handling edge connectors. Using the facilities of the layout menu, you can successfully construct the edge connector that will meet the specifications required by that program.

Select the *CONNECT* item on the main layout menu. The menu will change to the main *CONNECT* menu.

Select:
EDGE CONNECT
Top
Left
Bottom
Right
HEADER
Up
Down
To Right
To Left
-LAYOUT-

Ignore the entries for HEADER; these will be covered in the next section. Select the edge on the board on which the connector is to be constructed with the menu. This will make sure that the orientation is correct. The menu will be replaced with a menu asking for finger spacing.

TOP EDGE CON
100 mil
125 mil
156 mil

Select the spacing that you need. AutoCAD will ask for the point of insertion. Designate the location of pin #1 AT the edge of the board. The center of the pin will be at this point.

You will be prompted to enter the number of fingers (pins) in the connector. This will be in the prompt from the AutoCAD array command for number of rows or number of columns. Enter the number of fingers appearing on ONE side of the board. That's ONE HALF of the total number of fingers in the connector.

AutoCAD will draw the edge connector and return to the connector menu. The appearance of the edge connector will not be the same as it will appear in the routed artwork. The final artwork will have properly sized fingers and small traces leaving the fingers as routing targets.

Headers

Headers of several sizes can be constructed during the design of the board. The Auto-Board System II supports headers in three configurations:

- Single Row
- Double Row
- Staggered Double Row

The router has a special program for handling headers. Using the facilities of the layout menu, you can successfully construct the headers that will meet the specifications required by that program.

Select the *CONNECT* item on the main layout menu. The menu will change to the main *CONNECT* menu.

Select:
EDGE
CONNECT
Top
Left
Bottom
Right
HEADER
Up
Down
To Right
To Left
-LAYOUT-

Ignore the entries for *EDGE CONNECTOR*; these were covered in the previous section.

Select the direction in which the header will run. This direction is relative to pin #1. This will make sure that the orientation is correct. The menu will be replaced with a menu asking for the type of header.

To Right
1 Row
2 Row
Stagger
2 Row
HDR MENU

Select the type of header that you need. **AutoCAD** will ask for the point of insertion. Designate the location of pin #1.

You will be prompted to enter the number of pins in the header. This will be in the prompt from the **AutoCAD** array command for number of rows or number of columns. Enter the number of rows of pins in the header. That's **ONE HALF** of the total number of pins in the header for a double row header.

AutoCAD will draw the header and return to the connector menu.

Busses

One of the most frustrating operations in older versions of component placement software was the use of busses. This is now the most improved feature of **The Auto-Board System II**. You may specify buss runs with lines and polylines of actual width, instead of pulling and dragging an ambiguous shape around the screen. There is still work to be done to make the use of busses as simple as it should be. Using LISP will certainly help (not implemented yet) to simplify the operation some more.

Select **BUSS** from the main layout menu. This menu will be replaced by the main buss menu.

BUSS

 SELECT
 BY
 LINE
 TYPE

 POLYLINE
 Sldr BUS
 Part BUS

 LINE
 Sldr BUS
 Part BUS

 -LAYOUT-

Signal
 GROUND
 +5vdc
 +12vdc
 -12vdc
 Vdd
 Vcc
 Vss
 CUSTOM
 BUS

Busses can be built from either lines or polylines. When the polyline is used, the width of the buss trace is extracted from the width of the polyline. This makes the representation of the buss on the layout drawing to be very accurate.

From the BUSS menu, select the LINETYPE for the buss and the surface on which it will run.

AutoCAD will insert a special block which will be used to determine the signal that will be carried by the buss. This block is a single point on the drawing. The attributes of the block will be used to determine the width of the buss trace (for lines), the name of the signal and the anchor point for the buss.

AutoCAD will prompt for the point of insertion. Select a point that lies at one end of the buss trace. The buss can have several segments. Find an endpoint.

AutoCAD will prompt for signal name and the menu on the screen will give you a selection of standard signal names. Select one of the standard names or *CUSTOM*.

Signal:
GROUND
+5vdc
+12vdc
-12vdc
Vdd
Vcc
Vss
CUSTOM
BUSS

The custom selection will allow you to enter the name of the signal from the keyboard. This is handy for naming data or address busses. **AutoCAD** will have the "SIGNAL NAME" prompt ready **BEFORE** you can choose the *CUSTOM* item. Don't fall for this ruse. Use the custom entry. If you enter the signal name without the menu selection, the buss sequence is stopped without drawing a line. The *CUSTOM* menu item gathers the name of the signal and continues with the sequence through drawing the buss on the screen.

If you have selected the *LINE* linetype, you will be prompted for a trace width. Enter the width in real inches (i.e., ".02") notation and not in mils.

With the entry of signal and width data (for lines), the selection will start the polyline or line sequence. You will be asked to enter a first point. Select the same point that you selected earlier. This is how the name of the signal can be associated with the polylines / lines being used to draw it.

Use the "W" option in polyline to set the width of the buss.

Draw the buss around the board as required.

4.3 Ending the Session

To write the files that the *LAYOUT* program will use to generate its portion of the netlist, you will have to select the *BOOKWORK* item on the main menu. The items on this menu that will be of interest to you are:

FINISH
EDGE OUT
END
GET OUT

4.3.1 Writing the Files

As stated above, the *EDGE OUT* item may have been used at the start of the session. This will generate a shorter DXF file than it would if you wait to the end of the session to perform this function.

The command that you are interested in most at this time is the *FINISH* command which will send information about parts placement to the attribute TXT file. As in other commands and menu items in the system, all of the work has been done for you behind the menu selection. Select the *FINISH* command to generate the attribute file with positioning information.

The *GET OUT* and *END* items are there for your convenience. They are menu items for QUIT and END, respectively. If you have selected *EDGE OUT* and *FINISH* prior to this choice, you are ready to end the session. Selecting *END* will cause a script file to be started that saves the drawing and exits AutoCAD automatically. It even has the "0" required to leave the main menu of AutoCAD. With *EDGE OUT* and *FINISH* commands out of the way, you have enough information stored for use by the router. Neither QUIT nor END will interfere with the existence of those files.

The choice that you will have to make at this time is whether to save the DWG file or throw it away. The question that is necessary to ask is whether you may want to use the layout drawing again. One of the features of the router is its ability to generate a ratsnest drawing of your board. With this feature, you will be returned to your layout drawing and will be able to move parts around to clear up conflicts in the ratsnest. With a corrected layout, you would extract the DXF file with the *EDGE OUT* command and the new part locations with the *FINISH* item and proceed to route the board again with the cleaned up layout. Further details about generating the ratsnest are available in the next chapter.

4.4 Component Placement Software

The following section describes the functions of the software programs and batch streams that generate the netlist from your component placement.

The layout software is responsible for interpreting the DXF and TXT files that AutoCAD writes. These files contain information about the nature and content of the AutoCAD drawing. This information is extracted and rearranged to produce a netlist of the board that you have just laid out. This task is accomplished through two programs:

LAYOUT Netlist Generation

BUSCNV Buss Detection and Conversion

Their use is extremely simple but their execution should be controlled by a batch file to save you some typing and "sitting around" time. The batch file, BOARD.BAT, has been provided for you for this control function.

4.4.1 Netlist Generation

The LAYOUT program reads the attribute ("TXT") and DXF files to interpret part locations and board edges. The output is the part declarations of all parts found in the AutoCAD drawing and their locations. This program is activated by the following command line:

```
layout layout-name
```

4.4.2 Buss Detection and Conversion

The BUSCNV program reads the attribute ("TXT") and DXF files written by AutoCAD when you select *FINISH* and *EDGE OUT* on the menu. With positioning information supplied by the attribute file and line locations from the DXF file, BUSCNV writes the netlist entries for the busses found in the drawing. This program could be activated by the following command line:

```
buscnv layout-name
```

This program must be run after LAYOUT.

4.5 The Batch File Examined

The Component Placement (BOARD.BAT) batch file is provided to perform the sequence of operations of parts layout. The batch file covers the following operations:

1. Selecting the layout symbol library for AutoCAD. This is done with the "SET ACAD=" command. In the batch file, this is seen as:

```
echo off
set acad= \layout
```

2. Starting AutoCAD with initialization for a new drawing. Facilities of MS-DOS allow a batch file to check for the existence of a file. Depending upon whether a file exists or not, the operating system can allow certain logical decisions to be made. In the case of this batch file, the command to start AutoCAD is tailored to include a script file to request a new drawing and bring in the component placement menu. Old drawings don't require such initializations but another script file brings up the drawing editor automatically to keep things consistent.

The batch file does this with:

```
if exist %1-1.dwg goto old
acad %1-1 layout
goto join
:old
acad %1-1 old
:join
```

In the delivered batch file, there are a couple of other lines that are used when the ratsnest program is being used. These use "-R" notation to indicate the ratsnest file names. Please do not modify that sequence.

3. Starting the schematic capture software. Relying upon the check for existing files, the batch file runs the LAYOUT and BUSCNV programs. They are triggered by the existence of certain files written by AutoCAD. If these files are not found (meaning that the layout is probably not finished yet), the execution of the programs is by-passed.

The batch uses the following lines to perform this:

```
if exist %1-1.txt layout %1-1
if exist %1-1.dxf buscnv %1-1
```

4. Cleaning up the intermediate files from the session.

```
if exist %1-1.dxf del %1-1.dxf
if exist %1-1.txt del %1-1.txt
```

The "if exist" passages aren't really necessary since MS-DOS can hardly delete a file that doesn't exist. However, we have found that the user community feels so much better when the message *File not found* does not appear on the screen.

There are no conventions for naming schematic drawings. However, the batch file that is delivered to you does promote the use of the "-L" suffix for schematic drawings. When the batch file is started by entering:

`board test`

the internal commands append the "-L" to the name of the project, "TEST". You may remove the suffix from the batch file commands if you wish to adopt a different convention.

While we are on the subject of batch streams, one other batch file is included on your distribution disks. It is on the layout disk and is called "PARTS.BAT". This stream should be started with LAYOUT (the layout library) directory as the current directory. Running this batch stream will cause all component drawings to be erased and re-generated from their respective script files. Use this batch stream whenever you have modified a part in the library and are not sure about replacing just one element. This is the batch stream that we use to generate the library from time to time.

Chapter 5

The Auto-Router

5.1 Introduction

This chapter covers the use and control of the auto-routing module of **The Auto-Board System II**. This module contains several programs that interpret and transform the netlists that were developed in the schematic capture and component placement phases into the artwork of the routed board.

It is not vital that you understand the operation of these programs, so these details are not covered in this manual. However, information concerning the various settings that you can use to control them is offered so that you can get the most out of this software.

5.2 The Batch Files

All operations of the auto-router are performed through the use of batch streams. These batch streams are contained in files with the extension ".BAT". They are ASCII files and may be modified to suit your needs with a text editor or word processor. The batch files are located in the **SOFTWEST** directory and are constructed to follow the directory and file structures introduced in the installation of the package. If you're not up-to-speed with MS-DOS, please don't modify the batch streams.

There are two batch streams delivered with your software. The first is **RTE**, the auto-router control file. This batch file controls the sequence in which programs are executed to produce a routed board. The other batch stream is **RAT**. This batch stream controls the programs to produce a ratsnest representation of your board.

5.2.1 Using the RTE Batch Stream

The RTE command is used to produce the AutoCAD script file that represents the routed artwork of your circuit. When this batch stream is copied from the distribution diskette to the SOFTWEST directory, it is in a form that complements the batch streams for schematic capture and component placement. The netlist from the component placement will be expected to be using the "-L" notation and the schematic capture netlist having the "-S" notation.

To start the RTE batch file, enter:

```
rte test
```

where TEST may be the name of the project on which you are working. The rest of the operation is automatic. At the end of the batch stream, there is a command to start the **AutoCAD** program for displaying the results of the routing.

5.2.2 Using the RAT Batch Stream

The RAT command is used to activate the same sequence of software that the RTE command does, except that a ratsnest (point-to-point) drawing of the circuit is produced instead.

To start the RAT batch file, enter:

```
rat test
```

where TEST is the name of the board or project.

With the introduction of **AutoCAD 2.5** and its STRETCH command, an important feature for the development of printed circuit CAD was made available, "rubber-banding". It is not necessary to have **AutoCAD 2.5** to use the ratsnest operation; it is necessary to have **AutoCAD 2.5** with ADE-3 to use the STRETCH command. Using the STRETCH command, you may move components about the printed circuit board and any lines attached to the component will track with the component. To capitalize on this new feature, the ratsnest function of **The Auto-Board System II** is tied to the component placement module.

Here is what happens. When the RAT command is used with **The Auto-Board System II**, a special ratsnest program is called to create **AutoCAD** commands to draw the ratsnest. The RAT command then starts the BOARD batch stream. Don't worry, it's automatic. **AutoCAD** displays the drawing from component placement and proceeds to include the ratsnest lines that were created previously. Now you are in a

position to MOVE components around on the screen with the ratsnest lines following. You can un-tangle the ratsnest to get a neat and efficient layout.

This is how you use STRETCH. The **AutoCAD** STRETCH command is similar to MOVE when it is used with this application. With STRETCH, you form an **AutoCAD** "crossing" window (see **AutoCAD** reference for more information) around the component that you wish to move and slide the window (represented as a phantom drawing of the part) around the screen until the ratsnest lines show more efficient placement. That one is a judgement call that you will have to make.

Now the particulars:

1. Enter STRETCH from the keyboard, tablet menu or screen menu. Your configuration may vary.
2. **AutoCAD** will prompt you as to how you are going to select the components to stretch. Enter C for crossing. This is like the regular **AutoCAD** window except that it also selects all lines crossing the boundaries of the window. In this case, the crossing lines would be the ratsnest lines that you want to move with the component.
3. Form a window around the component to be moved. Choose a point to the lower left of the component that is close enough so that other components are not in the window, but wide enough to get the ends of the connecting ratsnest lines within the window.
4. Pick the upper right corner of the window to meet the same requirements.
5. Press the ENTER key on the keyboard, pointing device or tablet to terminate the selection of components.
6. **AutoCAD** will prompt for a BASE point. Move the cursor to pin 1 of the component and select it.
7. **AutoCAD** will prompt for a NEW point. Move the cursor to the new base point. The phantom drawing of the component and all connected ratsnest lines will follow. Press the ENTER key on the keyboard, digitizer or tablet to set the final position of the component.
8. Continue moving the components around until a pleasing layout is formed.

Since you will probably going to be saving (END command) this new layout, it may be convenient to use the CLEAN screen menu item to remove the ratsnest lines from your drawing. The CLEAN command is in the BOOKWORK menu. Use it before selecting FINISH or EDGE OUT.

If you will recall, the RAT batch stream started the BOARD batch stream, so if you used the FINISH command in the component placement menu, the LAYOUT program will startup and produce the new netlist.

Use the RTE batch stream to route the board with the improved layout.

5.3 Introduction to Programs

The following sections deal with descriptions of the options that can be used with the various programs in the auto-routing module. It is not essential that you read this section unless you are wanting to change some of the characteristics of these programs.

For experienced The Auto-Board System II users, we do suggest that you scan the section on the router configuration file. This will give you an idea of what kind of settings can be used with that program.

5.3.1 The NET Program

The Netlist Interpreter is the program that reads the information produced by the other modules (schematic capture and component placement) in the system or modified by you with a text editor. Its task is to translate the information into the forms required by the various programs in the auto-routing module. NET divides the information into various files. When the batch streams sense that these files are present, it activates the specialty software to handle them.

Using the NET Program

The Netlist Interpreter program should be run from the batch stream. However, being quite flexible, the program can be configured in a number of ways. You could modify the batch stream to take advantage of this fact.

NOTE: If you choose to modify a batch file (or any other file), make a copy of the original into a second file and modify the second file.

The Netlist Interpreter program is called NET. There are several parameters (names of files to be used by the program) that may be given to the program. These are:

1. The name of the board that is to be produced. This would most likely be the name that has been used throughout the system so far.

2. The name of the first netlist. The NET program can read several netlist files. This first netlist file is usually the one with component declarations and locations. This data is required by the NET program before any connections can be handled.

The RTE batch stream doctors the name of the board with "-L" to provide this parameter.

3. The name of the second (and subsequent file names following) netlist file. The NET program continues to pick up the names of the netlist files to read the contents. This netlist would usually be the file of nets from the schematic capture.

The RTE batch stream doctors the name of the board with "-S" to provide this parameter.

If there are several sheets of schematic drawings, then there will be several netlists. If you need to do this, modify the RTE batch file in the following manner.

Change:

```
NET %1 %1-L %1-S
```

to:

```
NET %1 %2 %3 %4 %5 %6 %7 %8 %9
```

To use this new batch stream (let's call it RTE2), enter:

```
rte2 test test-L test-S1 test-S2 test-S3
```

where test-S1, test-S2 and test-S3 were the names assigned to the netlists from the three schematic drawings that produced the netlists. The MS-DOS "REName" command can be used to change the names of files.

The Netlist Interpreter produces a parts listing on the screen as it encounters components in the layout netlist. These lines and others produced by the program (all or nothing) can be diverted to a file by adding the following to the END of the NET command line in the RTE and/or RAT batch streams:

```
> %1.lst
```

This is MS-DOS redirection of I/O and it causes output destined for the screen to be written to a disk file instead.

5.3.2 The Routing Program (ROUTE)

The purpose of the auto-routing program is to read in the trace-end pairs created by the earlier intermediate programs and build traces to form that circuit.

The router program is quite substantial in size (approximately 300K when linked, but is subsequently shrunk to 75K). The program uses even more memory (gathered from the operating system) during its runtime so it helps to have 640K available.

The auto-router will exercise very method that it has at its disposal to make a connection. The router uses a form of routing known as PROBE routing. It seeks to keep the vertically oriented tracks on one side of the board and the horizontal tracks on the opposite. Certain circumstances, however, call for bending these guidelines a little bit to facilitate a completed route. The router in *The Auto-Board System II* can be switched to become a maze router (in the works), breaking most of the rules that are so rigidly set for the probe router (important design rule criteria is not broken). Please report those instances when it uses methods that are not supposed to be at its disposal.

The router may fail to make a connection when a connect has not been made after the router has tried all reasonable paths between the two end points; run out of time; or hit a trace/via limit. A message is written to the screen to warn of trace connect failure.

A trace limit is in effect that limits the number of trace segments that may be attempted before calling the circuit a failure. This trace limit is set to six trace segments. You may change this value through one of the program parameters.

The number of attempted trace segments is progressive. The router will attempt to connect two points with three traces before it will attempt a connection with four traces. In this way, the most efficient route will be found. There is a parameter in the configuration file for overriding this value.

Sometimes it may seem reasonable to suspect that if a trace is not established within some time period that it is not likely to produce a very elegant trace. To prevent wild goose chases, the router also has a time limit that it will spend on any one connection. The default time limit is 10 minutes. You may set the time limit to any value that you wish with one of the program parameters. You may also disable the timer with the same parameter. We have found that time limits in the 5 to 10 minute range are optimum for a good mix of completion of traces and getting the board done within a reasonable time frame.

As the routing progresses, the points that the router is attempting to connect are displayed on the console screen. This is to give you the comfortable feeling that everything is still working. If the router should throw in the towel and fail in connecting two points, a message stating the cause of failure (*OUT OF TIME* or

CONNECT FAILURE) and the endpoints is written to the console screen.

These failures will show up on the **AutoCAD** drawing as straight lines (connecting the endpoints of the missed trace) on a **RATSNEST** layer. You can use the coordinates of these lines to manually connect the endpoints with **AutoCAD**.

All coordinates on the screen are given in mils in x,y format.

The router is an iterative router. After all endpoints have been attempted and an optimization pass has been made, the router will attempt to route the traces that failed on the previous pass. The cycle of routing and optimizing will continue until all traces are completed or until a pass is all failures.

It has been our experience that you can get very tired waiting for the router if it is successful only once in a pass of failures, prompting the router to try them all over again. The ESC key (escape) can be used to force the cancellation of a single route. Pressing the key for every trace will lead to the completion of the route (with all failures).

Router Options

The **Auto-Board System II** auto-router finds its options in one of three configuration files. These files are used to set the design rules that the router is to follow in its work. The **Auto-Board System II** options include:

- 1STEP** Routing is predominately done with vertical traces occupying on side of the board and the horizontal traces on the other. This option sets a limit to the routing that may be done on the "wrong" side to get around obstacles (feed-thrus or traces).
- HOME** This option sets a limit to the routing that may be done on the "wrong" side to get from a current position to the target coordinate (home).
- RIP** The rip-up and re-route optimizer has a threshold setting for the complexity of the trace that is a candidate for rip-up. The value of this parameter sets the minimum complexity. The measurement is the number of trace segments associated with a connection. Good results can be obtained by setting this parameter to 4.
- POLY** This parameter sets the output mode of router. If the value for this parameter is "YES" (or simply "Y"), the output generator will cause traces to be drawn as **AutoCAD** polylines (PLINE).
- STEP** This parameter sets the search step for the router. As the router looks for an open area to expand and route, it may search back and forth to get around

objects. The size of this search step can be critical in finding the first available path. Setting the value low can cause a slow, yet compact routing. Setting the value higher can cause a quicker route which may be somewhat sloppy by missing some obvious paths. This happens when the step size moves the search past a narrow, yet acceptable window. Good results have been obtained when this value is close to one-half of the default trace size.

NL This parameter sets the maximum number of layers to be routed. This value may be set to 1 or 2 on the routers that do not have the multi-layer extension.

TS This parameter represents the default trace spacing design rule. This value is measured from edge of trace to edge of trace. This parameter is overridden by the trace spacing value in the netlist.

PS This parameter sets the value of the pad spacing design rule. This value represents the allowable minimum distance between pads and feed-thrus during the routing process. Setting this value too low will create boards that are difficult to manufacture; while setting it too high will cause a board to lack the density that this router was written to produce.

TPS This parameter represents the default trace-to-pad spacing design rule. This value is measured from the edge of the trace to the edge of the pad or feed-thru.

TW This parameter sets the default trace width design rule. This parameter is overridden by the trace width value in the netlist.

FEED This parameter sets the default value of the pad (feed thru) size. This value represents the diameter of feedthrus that are inserted during routing. The size is also carried to the output when the feed-thrus are inserted. The name of the inserted feed-thru is derived from this number. If FEED is set to 35, then a drawing of *FEED35* will be inserted for the feed-thrus. Only *FEED40* and *FEED50* are supplied with the system. You will have to supply any other feed-thru drawings.

VIA Same as *FEED*.

FL (not implemented) This parameter sets the limit to feed-thrus allowed in a trace when the maze router is being used. Setting this value low will make the router remain on a single layer longer. Setting this value high will allow a better completion rate, but it will generate excessive feed-thrus.

BLIND In the multi-layer router, setting this parameter to "YES" (or simply "Y") will allow the use of blind vias. Blind vias (feed-thrus) are vias between adjacent routing layers (doesn't require putting a hole through all board layers to make a simple side swap). This is available only on the multi-layer routers.

RAT Setting this parameter to "NO" (or simply "N") will disable the output of ratsnest lines from the output of the router.

OPT This parameter represents the optimizer switches for the router. Specific use of this parameter will be covered later. There are four planned optimizers (two are not implemented at this time).

- Feed-thru reduction
- Rip-up and re-route
- 45° cuts at corners (100 unit maximum)
(not implemented)
- 45° cuts at corners (no maximum)
(not implemented)

SEG This parameter sets the maximum limit to the number of trace segments that may be used to complete a route. This value can help determine the number of feed-thrus that will be on a board. Every trace segment represents a turning of the trace to find its way to the target and probably will cause a feed-thru to be placed at that point.

START This parameter is somewhat the opposite of *SEG*. This value sets the starting number of allowable traces to be used to complete a route. When the router runs, it first allows some beginning number of trace segments to complete a route. If that number is found to allow too few segments, the allowable trace limit is incremented until a trace is completed or the limit set by *SEG* is reached. Setting this value too low causes extra time to be spent on worthless attempts to make a connection; but it does produce the cleanest routing. Setting this value too high and the completion rate will increase; but the elegance of the routing is very low. Setting this parameter to 3 seems to work the best. Setting it to 5 can create some very strange patterns (including loop-the-loops!).

TIME This parameter sets the time limit that the router may use to complete a trace. It is possible to set a different time limit for every pass of the router.

LAYER This parameter sets the layer pairs on which the router will operate. Specific details on the use of this parameter will be covered later. Although this option is primarily for multi-layer routing, this parameter can be used on the two-layer routers. For these routers, the values can be set to allow single sided routing or double sided routing.

PWR (not implemented) This parameter sets a dedicated layer for the routing of any trace with a recognized power signal number.

GND (not implemented) This parameter sets a dedicated layer for the routing of any trace with the ground signal number.

- BKUP** (not implemented) This parameter sets the time in minutes between file backups of the routed board.
- STYLE** (not implemented) This parameter sets the routing style. Setting the value to "P" causes the probe router to run. Setting the value to "M" lets the maze router operate. These values are set for each cycle of the router.
- MAZE** (not implemented) This parameter controls the layer on which the maze router may search.

5.3.3 The Configuration Files

Design rule checking values are assigned through a configuration file. Actually there are three configuration files. This allows some levels of control in establishing design rules. Suppose that your working drawing directory is PCB and the routed drawing is TEST. The following are configuration file names are valid.

\SOFTWEST\ROUTE.CFG This configuration file would always have this name. It is the first of the configuration files to be read. It is an optional file. It would contain NORMAL operational configuration settings.

\PCB\ROUTE.CFG This configuration file, if it is found by the program, is read next. Its contents would override the settings of the earlier configuration. This allows you to set some specific values for a PROJECT or group of boards. This file is optional.

\PCB\test.CFG This configuration file, if it is found by the program, is read last. Its contents would override the settings of the earlier configuration. This allows you to set some specific values for a board.

All of these files are optional. If they do not exist, the default values of the program will be used.

Construction of the Configuration File

The configuration file consists of a list of router options and their assigned values. There is no particular order in which they are to appear. There are two formats to the entries, though.

1. *name = value*
2. *name = [value value value ...]*

5.3. INTRODUCTION TO PROGRAMS

79

The spaces around the "=", "[", and "]" are required.

Form #1 is used to set a global value for routing. All entries can use this form. The value assigned is used during all passes of the router through the connect lists.

Form #2 is used to change the design rule value from one pass to the next. This form is limited to certain parameters. These parameters are:

- OPT
- SEG
- START
- TIME
- LAYER
- STYLE
- MAZE

Optimizer options The optimizer switches have four positions and two settings (for the present). The switch is written as a four digit value with each digit controlling one of the optimizers.

The four positions are:

1. Feed-Thru Reduction (left-most digit)
2. Rip-up and Re-route
3. Small 45° cuts
4. Large 45° cuts (right-most digit)

The values are:

- 0 Not active
- 1 Active at end of routing cycle
- 2 Active after trace (not implemented)
- 3 Both 1 and 2 (not implemented)

For example, "1100" is the switch setting to activate the feed-thru reduction and the rip-up optimizers. The other optimizers are off.

There is no rip-up *after trace* option.

These settings can have different values for each cycle of the router.

Layer options The LAYER parameter is for directing the pairs of layers that are to be used in routing the board. The value is a pair of numbers, separated by a comma. The first number is the vertically oriented layer and the second number is the horizontal layer. These values may change from router pass to pass.

For example:

```
LAYER = [ 1,2 3,4 1,4 3,2 ]
```

- Use layers 1 and 2 on the 1st pass
- Use layers 3 and 4 on the 2nd pass
- Use layers 1 and 4 on the 3rd pass
- Use layers 3 and 2 on the 4th pass

Notice that odd numbers are first in the routing pairs. This is so stay consistent in the numbering. There is no requirement that this numbering be maintained.

Setting a layer pair to the same layer will result in single-sided routing. For example:

```
LAYER = [ 1,1 ]
```

is a single sided router. And,

```
LAYER = [ 1,1 2,2 3,3 4,4 ]
```

creates a multi-layer board with no vias (other than inserted device pads).

Example Configuration File Here is a typical configuration file:

```
STEP = 5
HOME = 100
RIP = 4
```

```

PS = 20
TPS = 10
TS = 10
TW = 10
VIA = 40
lstep = 80
opt = [ 1111 1111 1111 1111 ]
nl = 4
time = [ 2 2 3 3 4 4 5 ]
layer = [ 1,2 3,4 1,4 3,2 2,1 ]
seg = [ 8 8 10 10 ]
start = 3
blind = Y

```

Figure reffig5-1 reviews the options that are available on each of the routers. Some options may be available, but not fully operational, depending upon the type of router being used. For instance, the *NL* option can be used on the 50 mil grid router, but the maximum value is 2 and not 16.

5.4 Finishing a the Artwork

The library parts for the router are found in the ROUTER directory. The netlist reader program generates AutoCAD INSERT commands to place the required parts into the drawing from this library. These parts will be located at the position and in the orientation that you stated in the netlist.

The router generated LINE, TRACE or PLINE commands to create the artwork for your drawing.

There are some things that you will have to do to prepare the drawing for final artwork production.

One of the first things will be to complete the routes that the router was unable to connect. These routes are indicated by ratsnest lines in a contrasting color on your drawing screen. Use the AutoCAD editing commands to move interfering traces out of the way.

You can make a drill list from the drawing. Do this before you fill the pads on the drawing. The pads on the drawing are blocks and carry an attribute of drill hole diameter. The attribute is named *HOLESIZE*. These drill sizes and the X,Y coordinates of the hole can be extracted from the drawing. There is an attribute template file on the system that will set up the output. The following steps are used to gather this information.

Option	Range	Grid	Fine Line	ARE-1	ARE-2	ARE-1 ARE-2
1STEP	0 .. ?		•	•	•	•
HOME	0 .. ?		•	•	•	•
RIP	0 .. ?		•	•	•	•
POLY	Y or N	•	•	•	•	•
STEP	1 .. ?		•	•	•	•
PWR	0 .. 16		•	•	•	•
GND	0 .. 16		•	•	•	•
NL	1 .. 16		•	•	•	•
BKUP	0 .. ?	•	•	•	•	•
TS	1 .. ?		•	•	•	•
TPS	1 .. ?		•	•	•	•
PS	1 .. ?		•	•	•	•
TW	1 .. ?	•	•	•	•	•
FEED	1 .. ?		•	•	•	•
VIA	1 .. ?		•	•	•	•
FL	0 .. ?		•	•	•	•
BLIND	Y or N		•	•	•	•
RAT	Y or N	•	•	•	•	•
OPT		•	•	•	•	•
SEG	1 .. ?	•	•	•	•	•
START	1 .. ?	•	•	•	•	•
TIME	0 .. ?	•	•	•	•	•
STYLE	P or M		•	•	•	•
MAZE	1 .. 16		•	•	•	•
LAYER		•	•	•	•	•

Figure 5.1: Router Options

- Enter the **AutoCAD** attribute extract command : *ATTEXT*
- Select the "Comma Delimited" format; enter "C". This is the default.
- Enter the name of the extraction template file: *DRILL*
- Enter the name of the output file (NOT *DRILL*). The default is the name of the drawing.
- A message will appear and **AutoCAD** will return to the drawing editor.
- The output file will have the file extension of ".TXT". It will be in your working directory. Some subsequent processing may be required to drive a drilling machine.

One of the last things that you will have to do is to fill in the pads so that they are ready for plotting. Up to this time, the pads have been represented as circles so that drawing speed could be maintained. Filled pads take forever (or least it seems like forever) to redraw. To aid in filling the pads, we have used *BLOCKS* for the pads. Blocks may be substituted for other blocks in **AutoCAD**. So we will substitute filled pads for the circles.

Before this substitution can occur, the filled pad block must be built. This is something that you will have to do for yourself to meet your own needs. It isn't difficult. Here's what you do:

- Start an **AutoCAD** session with a new drawing. Its name doesn't matter, but let's call it, *FPAD50*.
- Set the limits so that you can see what you are doing (a one half inch square is a good size).
- Zoom into the limits.
- Create some new layers with *LAYER NEW*. Create these layers:
 1. PADS
 2. MASK
 3. DRILL
 4. PWR
 5. GND
- Set the layer to PADS with *LAYER SET PADS*.
- Create a donut or square pad (or whatever you want) with the center at 0,0.

- Set the layer to DRILL and draw a drill target.
- Set the layer to MASK and draw the circle for a solder mask.
- Set the layer to PWR and draw an isolation pad for the power plane.
- Set the layer to GND and draw an isolation pad for the ground plane.
- Save the drawing by ending the session with *END* or use *WBLOCK* to write the completed block to disk.
- If you used *WBLOCK*, build some other pads for headers, feedthrus, large resistor pads, etc.
- Use *QUIT* if you used *WBLOCK* to write the parts to the disk.

The names of the pads that you will encounter with The Auto-Board System II will be:

- RND_PAD (IC pads)
- SQR_PAD (IC pads)
- FEED40 (Supplied Feedthru)
- PAD_40 (Test points)
- PAD_50 (Headers and test points)
- PAD_60 (Resistors and Capacitors)
- PAD_70 (Resistors and Capacitors)
- SMD_PAD (Surface Mounted Parts)
- PLCC_PAD (Flat Pack and Chip Carrier Devices)

You have the ability within AutoCAD to exchange these blocks for blocks with filled pads, donuts or (if you wish) stars. Create a finished pad with AutoCAD as described above and place the drawing in a file within the ROUTER library directory. Then a single command to AutoCAD will make a global change. Use the command:

```
insert rnd_pad=FPAD50
```

to make the change. This command will replace all of the designated pads with the new filled pad.

Chapter 6

Custom Parts - Schematic Symbols

6.1 Introduction

One of the goals in writing The Auto-Board System II was to make it easy to add new parts. In the original form of The Auto-Router, it was possible to extend the parts library, but it certainly wasn't easy. In that early release, it was necessary to write additional software to recognize new groups of parts. With the introduction of The Auto-Board System version 1.3, all parts and symbols were defined outside of the software in a user maintained database. With the database, the definition of a 16 pin IC (supplied with the system) is no more and no less standard of a part than a custom transformer that the user may have added. Both appear the same to the system.

This chapter explains the mechanism for adding a schematic symbol to the schematic capture module of The Auto-Board System II. The process is not difficult. However, complex parts should not be added until the process is understood and practiced. Please be familiar with the following concepts before attempting to create a symbol:

- AutoCAD command structure
- AutoCAD attributes
- AutoCAD script files
- AutoCAD structure imposed by The Auto-Board System II
- Text editing

Get to know your computer and its software before starting on this task.

In general, the steps to create a custom schematic symbol are

1. Design the symbol on paper
2. Assign the appropriate attributes
3. Generate the AutoCAD drawing in blocks
4. Create database information
5. Create or update database

The following sections describe the creation of a unique schematic symbol.

6.2 Schematic Symbol Creation

The first thing that you will have to do in the creation of a part is to decide what it is that you want to create. That sounds stupid, but without some planning, the task can be confusing and frustrating. It doesn't have to be. One of the biggest mistakes for beginning computer programmers is to start writing a program without benefit of deciding what it is that the program is to do and how it is going to accomplish it. In a similar manner, to create a schematic symbol without some think-and-design time can lead to a genuine mess.

Design Design Design

Don't do the following steps with AutoCAD.

- Decide what you want to do.
- Draw it on paper.
- Play with it.
- Make it look like what you want.
- Label the leads

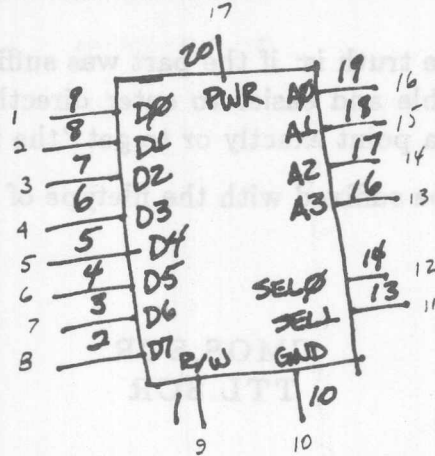


Figure 6.1: Hand Drawn Symbol Design

1. Number the leads sequentially from 1 to whatever (number of leads on the drawing). This numbering should be somewhat logical in its sequence.
2. Assign the leads with the number of the corresponding pin on the physical device. These numbers will be used when the attributes are generated.
3. Assign the leads with functional reminders

Figure 6.2 is a sample drawing. Obviously, this is NOT an AutoCAD drawing. It is a design. A freehand sketch. Notice that the leads are lightly numbered sequentially around the outside of the drawing. These are the drawing pin numbers. The heavier numbers correspond to the pins on the physical device that the symbol represents.

We will use the upper left corner of the symbol as the insertion point and the origin of the drawing.

6.3 AutoCAD Script Preparation

There are two ways in which the symbols can be created with AutoCAD. The obvious method is to use AutoCAD to draw and label the symbol. This is probably quite satisfactory for the creation of one symbol. However, when several similar parts are being drawn, we have found that the script file method to be faster to produce and easier to modify. When there are several drawing that are the same, it is a snap to copy sections of script instructions (to be used as templates) to quickly create several parts.

The script file can be built with a text editor or word processor. The contents of the file is simply the commands that you would type into AutoCAD to manipulate the drawing. You might argue that it is easier to use a digitizer to mark out the various positions on the screen than it is to figure out the coordinates and enter

them into a script file. The truth is: if the part was sufficiently described on paper, the coordinates are available and easier to enter directly than to move a digitizer puck around trying to hit a point exactly or to get "the right look".

Script filenames must be suffixed with the filetype of ".SCR". For example,

CMOS.SCR
TTL.SCR

6.3.1 Initialize AutoCAD

Like a normal AutoCAD session, the script file will contain some initialization commands.

In the passages that follow, the script command will be shown on the left and an explanation on the right.

1	begin a new drawing
x	dummy name
limits -2,-2 2,2	set some limits
zoom a	look at it
layer new parts	create PARTS layer
	required blank line
layer set parts	use PARTS layer
	required blank line

When you create the script file, you will not need to put in the comments. Be aware that there is a blank line after the LAYER commands so that AutoCAD will leave the LAYER operation.

Those steps do not have to be performed again in the session unless there is a change in the drawing environment.

6.3.2 Draw the Symbol

Now it is time to draw the outline of the figure that we add to the symbol library.

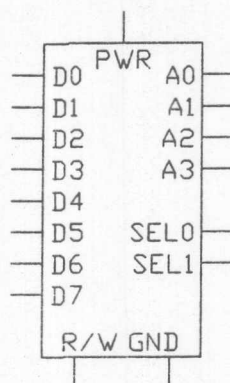


Figure 6.2: Outline of the Schematic Symbol

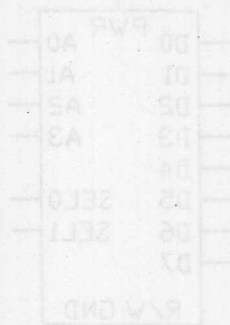
line 0,0 .5,0 .5,-1 0,-1 0,0	symbol outline
	required blank line
line -.1,-.1 0,-.1	lead for "D0"
	required blank line
array 1 r 8 1 -.1	draw "D0" thru "D7"
line .5,-.1 .6,-.1	lead for "A0"
	required blank line
array 1 r 4 1 -.1	draw "A0" thru "A3"
line .5,-.6 .6,-.6	lead for "SEL0"
	required blank line
line .5,-.7 .6,-.7	lead for "SEL1"
	required blank line
line .1,-1 .1,-1.1	lead for "R/W"
	required blank line
line .4,-1 .4,-1.1	lead for "GND"
	required blank line
line .25,0 .25,.1	lead for "PWR"
	required blank line

Next comes the text inside of the part to label the pins. These tags do not create any signals by themselves. *GND* does not automatically attach the pin to the GND net.

```

text .03,-.13 .055 0 D0      labels for leads
text .03,-.23 .055 0 D1
text .03,-.33 .055 0 D2
text .03,-.43 .055 0 D3
text .03,-.53 .055 0 D4
text .03,-.63 .055 0 D5
text .03,-.73 .055 0 D6
text .03,-.83 .055 0 D7
text r .47,-.13 .055 0 A0
text r .47,-.23 .055 0 A1
text r .47,-.33 .055 0 A2
text r .47,-.43 .055 0 A3
text r .47,-.63 .055 0 SELO
text r .47,-.73 .055 0 SEL1
text c .15,-.98 .055 0 R/W
text c .35,-.98 .055 0 GND
text c .25,-.08 .055 0 PWR

```



That completes the drawing of the basic outline and the associated text. There are two things that can be done at this time. One, the outline can be saved as a BLOCK. Or, the attributes could be added at this time. If the outline is generic in nature and will be used in several symbols, save the drawing as a block.

```

block memory 0,0 w -2,-2 2,2 save as block MEMORY
                                required blank line
redraw                          refresh the screen

```

Figure 6.2 shows what has been drawn so far.

6.3.3 Symbol Attributes

With this block drawing and its soon-to-be-added attributes, we can construct a schematic capture symbol. The attributes that will be added are:

- MARKER
- LIBRARY
- KEYWORD
- PINS
- PIN1 thru PINn
- DEVSPEC

- DEVNAME

MARKER This attribute specifies the use of the symbol. If the value of the **MARKER** attribute is "P", then the symbol represents a physical entity or component. An attribute of "S" indicates a signal name symbol which is used to assign a signal name to a symbol's lead or a line-group connecting several leads. This attribute is invisible and constant.

LIBRARY This attribute names the library database where additional information about the symbol can be found. This attribute is invisible and constant.

KEYWORD This attribute names the keyword that is used to enter the library database. This name selects a particular entry from the database. This attribute is invisible and constant.

PINS This attribute documents the number of leads on the symbol that may be connected in the schematic. It also specifies how many PINn attributes will be attached to the symbol. This attribute is invisible and constant.

PIN1 through PINn These attributes assign a lead on the symbolic drawing to a specific pin on the physical device. In the example, the leads of the symbol are numbered sequentially for reference points. These are the *light* numbers on the first drawing. These numbers correspond to the number in the PIN attribute NAME (i.e., PIN1, PIN34, PIN6). On each symbolic lead, a physical pin number was assigned with *HEAVY* numbers. In the example, the "D0" lead is attribute PIN1 (symbolic pin 1) and pin 9 on the real device. This device pin becomes the attribute VALUE. This attribute may be either visible or invisible. It is usually visible so that the pin numbering will document your drawing. This attribute may be either constant or variable. When the exact pin numbers are known, then it makes sense to set these values to constant. Sometimes, you will want the designer to designate pin numbers. For instance, in the case of a NAND gate, several gates appear in a single package. To use a single symbol to represent these several gates, you would let the pin assignments be variable. At the time of use, AutoCAD will prompt for the pin assignments for the specific implementation.

DEVSPEC This attribute is a documentation feature. This attribute is used to designate a symbol's device type. This attribute will be passed through the various phases of The Auto-Board System II until the value is set into the silk screen layer of the printed circuit board. This attribute is usually visible. The attribute can be constant when you know the device type being used. For instance, in a TTL or CMOS library, the device type is known when the library piece is constructed and the attribute is constant. On the other hand,

the symbol may be quite generic and a specific device type or value can be added at the time of usage. Then the attribute should be a variable.

DEVNAME This attribute is the unique symbolic name given to the particular part on the drawing. This will be the identifying value that will be used throughout The Auto-Board System II when referencing the part.

Your custom part should use all of these attributes. You may add other attributes, but *do not* add them to the schematic TXT template file (SC.TXT). This will confuse the software that interprets the attribute output files. We will have no pity for offenders in this area.

Refer to your **AutoCAD** User Reference (What a unique idea!) for more information about the structure of attribute commands. It is not the intent of this chapter to teach the use of attribute definitions. It is intended to guide you in using the attributes that are recognized by The Auto-Board System II.

For our example drawing, the following would be placed into the script file. This script assumes that the symbol outline has been saved as the block, MEMORY. It is possible that this portion of the script could be executed at a later time in the **AutoCAD** session than the script that created the symbol's outline. It is necessary that it be used in the same session.

```

insert memory 0,0 1 1 0      get copy of symbol outline
attdef i c                  define attribute invisible/constant
                             required blank line
MARKER P                    this is a PART
0,0 .01 0                  text characteristics (invisible)
attdef                      define attribute invisible/constant
                             required blank line
LIBRARY CUSTOM              find part in CUSTOM database
0,0 .01 0
attdef                      define attribute invisible/constant
                             required blank line
KEYWORD MEM                 this part is filed under "MEM"
0,0 .01 0

```

Once the initial and standard attributes are in place, continue with defining the individual pins. This is where the sequentially numbered pins are correlated with the physical pins of the component.

```

attdef          define attribute invisible/constant
                  required blank line
PINS 17          there are 17 leads on symbol
0,0 .01 0
attdef i        define attribute visible/constant
                  required blank line
PIN1 9           1st pin is physical pin 9
r -.03,-.08 .055 0 where to put it in drawing
attdef          define attribute visible/constant
                  required blank line
PIN2 8           2nd pin is physical pin 8
r -.03,-.18 .055 0 where to put it in drawing
attdef          define attribute visible/constant
                  required blank line
PIN3 7           3rd pin is physical pin 7
r -.03,-.28 .055 0 where to put it in drawing
attdef          define attribute visible/constant
                  required blank line
PIN4 6           4th pin is physical pin 6
r -.03,-.38 .055 0 where to put it in drawing
attdef          define attribute visible/constant
                  required blank line
PIN5 5           5th pin is physical pin 5
r -.03,-.48 .055 0 where to put it in drawing
attdef          define attribute visible/constant
                  required blank line
PIN6 4           6th pin is physical pin 4
r -.03,-.58 .055 0 where to put it in drawing
attdef          define attribute visible/constant
                  required blank line
PIN7 3           7th pin is physical pin 3
r -.03,-.68 .055 0 where to put it in drawing
attdef          define attribute visible/constant
                  required blank line
PIN8 2           8th pin is physical pin 2
r -.03,-.78 .055 0 where to put it in drawing

```

That takes care of half of the pin. Here are the other half of the pins.

```

attdef          define attribute visible/constant
                  required blank line
PIN9 1          9th pin is physical pin 1
r .12,-1.08 .055 0  where to put it in drawing
attdef          define attribute visible/constant
                  required blank line
PIN10 10        10th pin is physical pin 10
.38,-1.08 .055 0  where to put it in drawing
attdef          define attribute visible/constant
                  required blank line
PIN11 13        11th pin is physical pin 13
.53,-.68 .055 0  where to put it in drawing
attdef          define attribute visible/constant
                  required blank line
PIN12 14        12th pin is physical pin 14
.53,-.58 .055 0  where to put it in drawing
attdef          define attribute visible/constant
                  required blank line
PIN13 16        13th pin is physical pin 16
.53,-.38 .055 0  where to put it in drawing
attdef          define attribute visible/constant
                  required blank line
PIN14 17        14th pin is physical pin 17
.53,-.28 .055 0  where to put it in drawing
attdef          define attribute visible/constant
                  required blank line
PIN15 18        15th pin is physical pin 18
.53,-.18 .055 0  where to put it in drawing
attdef          define attribute visible/constant
                  required blank line
PIN16 19        16th pin is physical pin 19
.53,-.08 .055 0  where to put it in drawing
attdef          define attribute visible/constant
                  required blank line
PIN17 20        17th pin is physical pin 20
r .22,.03 .055 0  where to put it in drawing

```

The pin numbers are usually constant. If there is a need to tag a pin with a specific number, then the attribute tag for "i" or "c" needs to be toggled.

The next set of attributes are used for setting the device type and reference designators in the drawing. If this is a specific part, then the device type could be left as constant. One thing to remember when setting up these attributes is that

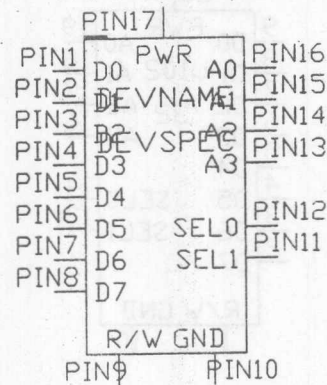


Figure 6.3: Drawing of Schematic Symbol with Attributes

the order in which they are entered is the reverse order in which the user will be prompted for a response.

attdef c	define attribute visible/variable
	required blank line
DEVSPEC	device type attribute
Enter Device Type	prompt
	default value
c .25,-.36 .06 0	text location
attdef	define attribute visible/variable
	required blank line
DEVNAME	device name
Enter Device Name	prompt
	no default
c .25,-.22 .06 0	
wblock mem	write symbol to file
	required blank line
0,0 w -2,-2 2,2	insertion point and write window
	required blank line
redraw	refresh screen

Figure 6.3 illustrates what AutoCAD will be displaying prior to the WBLOCK command.

And Figure 6.4 shows the appearance of the symbol when it is inserted into a drawing.

For further information, look at the script files (*.SCR) on the master distribution disks. Schematic symbols will be found in the FORMn.SCR and GENERICn.SCR files.

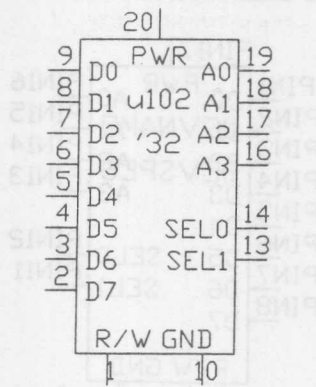


Figure 6.4: Schematic Symbol in a Drawing

This chapter has covered the steps for constructing a schematic symbol for The Auto-Board System II. The steps included:

- Design of symbol on paper
- Construction of outline with AutoCAD
- Addition of attributes

Chapter 7

Custom Parts – Layout & Router

Parts for the component placement and auto-routing phases can be created in much the same way as the schematic symbols. The differences in the procedures are minor. Where the purpose of the schematic symbol was to produce a symbol outline, the purpose of the layout and router component drawing is to produce a silk screen outline of the physical part and a pad master.

7.1 Drawing the Part

As stated, the layout and router drawing contains a silk screen outline of a component and a pad master drawing. To do this with **AutoCAD**, two layers are used. As you might guess, these layers are called "SILK" and "PADS".

First things first. Plan the part drawing on paper before attempting to use **AutoCAD** to draw it. Calculate the distances that are involved and where fine details (notches for ICs) are going to start and stop.

For the purposes of illustration, the following script files will generate the layout and router drawing for a 24 pin "skinny" DIP. The same drawing can be used for both the layout and the router modules.

First, set up the **AutoCAD** environment.

1	generate a new drawing
x	dummy name
limits -2,-4 2,2	set working area
zoom a	look at it
layer new pads,silk	build layers
	required blank line
layer set pads	start with pads
	required blank line
circle 0,0 .025	"check plot" pad
attdef i c	define attribute invisible/constant
	required blank line
holesize .03	drill diameter
0,0 .01 0	
block rnd_pad 0,0 w -1,-1 1,1	save it
	required blank line
line -.025,.025	square pad details
.025,.025	
.025,-.025	
-.025,-.025	
-.025,.025	
	required blank line
attdef	define attribute invisible/constant
	required blank line
holesize .03	drill diameter
0,0 .01 0	
block sqr_pad 0,0 w -1,-1 1,1	save it
	required blank line
redraw	

The definitions of RND_PAD and SQR_PAD are built in this manner so these "check plot" blocks can be substituted for "filled" drawing pads when the drawing is completed. Using filled pads could slow down the operation of AutoCAD when drawing are redrawn. Circles and squares are used for "check plot" purposes until final drawing are required.

Next, we will define the structure of an individual part. The first thing that we will define is the pad master layer.

<code>layer set pads</code>	activate the proper layer
	required blank line
<code>insert rnd_pad 0,0 1 1 0</code>	start with pin 1
<code>array 1 r 12 2 -.1 .3</code>	draw all pins
<code>erase w -.04,-.04 .04,.04</code>	remove round pin at pin 1
	required blank line
<code>insert sqr_pad 0,0 1 1 0</code>	use a square pad at pin 1

This sequence will draw 24 pins (12 rows with 2 columns in the ARRAY command) in the configuration of a slim (300 mil spacing) DIP. The round pad at pin 1 was erased and replaced with a square pad.

Now it is time to activate the SILK layer and draw the outline of the part. This layer will be left activated when the attributes are defined so that the text from the attributes will appear in the SILK layer.

<code>layer set silk</code>	activate SILK layer
	required blank line
<code>arc .18,.03 e .12,.03 a -180</code>	notch at top of IC
<code>line .12,.03</code>	
<code>.03,.03</code>	
<code>.03,-1.13</code>	
<code>.27,-1.13</code>	
<code>.27,.03</code>	
<code>.18,.03</code>	

required blank line

There is a blank line at the end of this sequence to terminate the LINE command.

Figure 7.1 shows the completed outline.

7.2 Layout Attributes

In the layout module, the attributes are used to extract information about the type and placement of parts. Attributes for the router module are covered in the next section.

The following sequence of attributes should be followed carefully. Invisible and constant values are not as negotiable as they were in the schematic capture section. The following are the names of the attributes:

- PREFIX

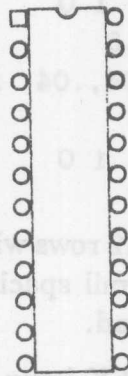


Figure 7.1: Component Outlines

- WIDTH
- SIDE
- NAME
- TYPE

PREFIX This attribute is a signal as to how the part should be handled. The value that is assigned to this attribute is the same as the netlist designator that follows the "=" (parts definition). See Appendix A for netlist information.

- A Array
- B Buss
- C Edge Connector
- F Feedthru
- H Header
- P Part

The PREFIX attribute carries with it the "=" needed by the netlist.

WIDTH At the current time, this attribute is used by BUSS for trace width of busses and ARRAY for the spacing between the ICs in a memory array.

SIDE This attribute is used to designate the side of the board that a component is mounted. This is used by the BUS command to determine the location of busses. It is also used with surface mounted parts to determine orientation of the part and its side.

NAME This attribute is the identifying name of the part. It MUST agree with the name of the part in the schematic diagram for successful routing. This name will be written to the silk screen layer of the routed board.

TYPE This attribute designates the device type of the part. The value of the part entered will be displayed on the silk screen layer of the PCB. The TYPE of the schematic symbol will override the value given here.

And the script file to use all of these attributes:

```
attdef i c                                define attribute invisible/constant
                                           required blank line

PREFIX
= P                                       this is a PART
0,0 .01 0
attdef i c                                define attribute visible/variable
                                           required blank line
TYPE                                     device specification
Enter Device Specification                prompt
                                           required blank line

c .15,-.15 .06 90
attdef                                     define attribute visible/variable
                                           required blank line
NAME                                     device name display this
                                           no default

c .15,.06 .06 0
wblock slim24
                                           required blank line
0,0 w -.4,-3.5 1.5,1                    insert point and write window
                                           required blank line
redraw                                   refresh screen
```

The remaining task is to add your new part to the LAYOUT menu. There is some area at the end of the menu dedicated to custom parts.

7.3 Router Attributes

In the router, the attributes are a good method of making sure that the lettering on the silk screen layer rotates with the part. Calculating text positions is difficult at best, so it easiest to let AutoCAD handle the positioning of text.

The following sequence of attributes should be followed carefully. Invisible and constant values are not as negotiable as they were in the schematic capture section. The following are the names of the attributes:

- DEVNAME

- OLDNAME

- DEVSPEC

DEVNAME This attribute is the identifying name of the part. This name is written to the silk screen layer of the routed board.

OLDNAME This attribute is for future use. It will be used for back annotation of device names that have been changed on the routed board and need to be updated on the schematic.

DEVSPEC This attribute designates the device type of the part. The value of the part will be displayed on the silk screen layer of the PCB.

And the script file to use all of these attributes:

attdef	define attribute visible/variable required blank line
DEVSPEC	device specification
Enter Device Specification	prompt required blank line
c .15,-.15 .06 90	
attdef i	define attribute invisible/variable required blank line
OLDNAME	don't want to show old name required blank line
0,0 .01 0	
attdef i	define attribute visible/variable required blank line
DEVNAME	device name display this no default
c .15,.06 .06 0	
wblock slim24	required blank line
0,0 w -.4,-3.5 1.5,1	insert point and write window required blank line
redraw	refresh screen

The part will be put into the router library with the WBLOCK command. It will be used as required by script file composed by The Auto-Board System II.

Chapter 8

The Database Files

This chapter continues the process of creating new symbols and parts for **The Auto-Board System II**. Unlike the complexities of dealing with different attributes for the various phases of the program, creating the database files is the practically identical for schematic capture and auto-routing. The component placement module does not use a database for the parts libraries.

The processes involved in this chapter can be used to build new database files or merely update the files that already exist on the system. In the case of the auto-router database (**NETLIB**), your only choice is to alter and update the database. For schematic capture, you may add symbols to existing libraries or create a special database and library for your own parts.

8.1 The Database Record

Each drawing in the schematic capture symbols library contains several attributes. One of those attributes is the **LIBRARY** attribute. This attribute is the name of the database file that contains the database record that corresponds with the drawing. There is also an attribute named **KEYWORD**. This attribute is a unique name of the part in the database. When the database is accessed with this keyword name, a database record is read. The record contains positioning information.

Likewise, in the auto-routing package, there is a database containing positioning and pad size information. There is no **LIBRARY** attribute because there is only one database, **NETLIB**. There are no **KEYWORD** attributes because the part identification is in the netlist (i.e., **IC16**, **RES18** or **TO5**) is the keyword into the database.

To add a part to either database, it is necessary to create a database record for that part. This is not a difficult task. You will need a text editor to do the job.

These are three section to the database record.

8.1.1 Symbol/Part Name

The first line of the record is the name of the part. This name corresponds to the KEYWORD attribute in the schematic capture drawing or the part designator in the netlist. This name should be 1 to 8 characters since it is also the filename of the drawing that will be inserted (in the case of the auto-router).

8.1.2 Pin Count

The second line of the records is the number of active leads/pins that are on the drawing or symbol. This number will correspond (and is somewhat redundant) to the PINS attribute in the schematic capture symbol.

8.1.3 Pin Offsets

The remaining fields in the record are the individual X and Y offsets of the various leads/pins on the part. Each line MUST contain:

1. The number of the pin
2. The X offset in thousandths
3. The Y offset in thousandths

And the following entries are *required* for an auto-router part:

1. The length of the pad (X dimension)
2. The width of the pad (Y dimension)

Use the maximum length and width for irregularly shaped pads. These values must represent the pad as it is centered at its insertion point on the drawing. Asymmetrical pads may cause routing interference. Idea: Take the farthest point from the mounting pad in the X and Y directions to calculate size for half of a pad, and double the values to get good clearance all around.

To build a database entry for the 24 pin "skinny dip" for the auto-router database, the record would look like this:

SLIM24

24				
1	0	0	50	40
2	0	-100	50	40
3	0	-200	50	40
4	0	-300	50	40
5	0	-400	50	40
6	0	-500	50	40
7	0	-600	50	40
8	0	-700	50	40
9	0	-800	50	40
10	0	-900	50	40
11	0	-1000	50	40
12	0	-1100	50	40
13	300	-1100	50	40
14	300	-1000	50	40
15	300	-900	50	40
16	300	-800	50	40
17	300	-700	50	40
18	300	-600	50	40
19	300	-500	50	40
20	300	-400	50	40
21	300	-300	50	40
22	300	-200	50	40
23	300	-100	50	40
24	300	0	50	40

If this part is to go into the auto-router database, it would have to be entered into the file called NETLIB.LIB. The ".LIB" suffix is the suffix understood by the database software as the source code of the parts descriptions.

For a schematic symbol, the coordinates would mark the end of the leads on the drawing. This would be the point that a connecting line would end to show that it is attached to the part. Of course, the last two fields in the record for pad size would not be present in the database record.

For a new database file for the schematic capture, the database record description would be placed in a file with the suffix of ".LIB" (such as "MYPARTS.LIB"). You can place several descriptions in each file. Look at GENERIC.LIB and NETLIB.LIB for more examples.

8.2 Build or Update the Database

Building or updating the database files is done in the same way. For an update, the files are automatically cleared and the database is literally re-built with the changes in place. It is therefore, not necessary to erase the old database when an update is required.

The **BILDLIB** program is used to construct the databases for schematic capture. **NETBILD** is used for the auto-router database. The **BILDLIB** program is started with the following command line:

```
bildlib name-of-database
```

For first-time builders: Make sure that the **SOFTWEST** directory is the current directory. If your databases are in this directory, make it the current directory before using the program. If you are keeping your databases in various directories, use that directory as the current directory for this operation.

Here is an example of using **BILDLIB**:

```
bildlib cmos
```

The **BILDLIB** program will create (if they do not exist) two files with the name of the database. The suffixes of these files will be ".IDX" and ".DTA". For the database called "MYPARTS", the files will be:

```
MYPARTS.IDX  
MYPARTS.DTA
```

The **BILDLIB** program will write a little message with the name of your file and a number. The number may not change from run to run, and it may not seem logical; but it is extremely important. Edit the **SOFTWEST.DIR** and add the information that was on the line: For example,

```
MYPARTS 121
```

may be the result of the **BILDLIB** program. You would then enter this line into the **SOFTWEST.DIR** file.

The **NETBILD** program builds the auto-router database. Since there is only one file that is ever used for the database, **NETLIB**, **NETBILD** will always read this file and needs no other names to run correctly. To run **NETBILD**, enter:

netbild

Again, **SOFTWEST** should be the current directory in this case. If you are doing exotic things, use the directory that contains the ".LIB" file as the current directory.

NETBILD also displays a name and number at its conclusion. Enter the name and number into the **NETLIB.RT** file in the current directory (**SOFTWEST** in this case).

Appendix A

The Netlist Language

A.1 Introduction

This appendix covers the structure and use of the netlist language.

The netlist is the method of defining the circuit that controls the routing processes. Although the function of the schematic capture and component placement modules is to produce the necessary files to run the router, it is good to understand the structure of the netlist files. It may be easier at times to modify the netlist instead of running all of the schematic capture and component placement programs to produce a small change. You would, of course, change the schematic diagrams and layout drawings to reflect any changes that you would make.

The netlist language has a very simple format. There are eight commands. The primary means of generating the netlist is through the schematic capture and layout packages in The Auto-Board System II. Since the netlist is in ASCII characters, it is available for editing and adjustments. As an overview, the commands in the netlist language are:

- **Part Definition ("=")** This command defines a component by assigning a device type (DIP, connector, resistor, bus, etc.) to a symbolic name.
- **Package Placement ("@")** This command defines the location and orientation of the component on the pcb.
- **Signal Name ("!")** This command names the current signal that is being assigned to the various component pins and leads.
- **Signal Connection ("*")** This command assigns the current signal to a specific component pin or lead.

- Device Type (“#”) This command labels a component with a device specification or value.
- Include File (“+”) This command allows another file of netlist commands to be included in the current stream of commands. This command is useful for combining several netlist files into a final design.
- Comment (“:”) The contents of the comment line are ignored by the netlist processor which allows you to document the contents of the netlist file. This represents a change from past versions.
- End of List (“.”) This command officially closes the processing of the netlist program.

A.2 Construction of the List

The entire file can be set up in a free-form fashion with several commands on a line permitted. Or you may spread a command over several lines. This may be helpful when defining the edges of the board.

In all of the following commands, several fields in the command structure may be designated by a single letter. However, the netlist processor will accept words. Only the first letter of the word will be used. This makes reading the list much easier for the human and no big deal for the processor.

A.2.1 Part Definition

By far the most powerful command in the netlist language is the part definition command, designated by the equals sign (“=”). This command defines every component (and then some) that will be used on the board. Since each component type is defined slightly differently, each will be described separately. The general form of the Part Definition command is:

```
= <part_type> <parameter> [parameter] ...
```

The part types and their specific command lines follow.

Circuit Component

This form is used to define most physical packages on the board. The notable exceptions are edge connectors, headers and ic arrays. The form of this command

is:

= P <component symbol> <component definition>

For example,

```
: declare resistor r1 to be a 1/4 watt resistor
: notice how the command "P" has been expanded to "Package"
= Package R1 res14
: declare ic U1 to have 16 pins
= P U1 IC16
```

The following parts are available with the database provided with the system. Expansion of the database is explained in Chapter 7.

IC8	IC14	IC16	IC18	IC20
IC22	IC24	IC28	IC40	IC64
FEED40	FEED50			
CAP100	CAP200	CAP250	CAP300	CAP500
CAP600	CAP800	CAP1000	CAP1500	CAP2000
CAP2200	CAP2500			
TO5	TO8	TO18	TO39	TO98
TO202S	TO202T	TO218	TO220S	TO220T
TO92	TO226			
DIODE				
RES18	RES14	RES12	RES1	RES2
SIP4	SIP6	SIP7	SIP8	SIP10
SIP11				
DB9P	DB9S	DB25S	DB25P	

Board Edge

This form of the definition command allows you to define the size and shape of the pcb. The definition is a series of adjacent corner coordinates of the board in X Y format measured in mils. Closure is required. The form of the command is:

= E <X> <Y> { <X> <Y> }

For example,

```

: define a tiny board
= E
0 0
0 1500
2000 1500
2000 0
0 0

```

A.2.2 Package Placement

This command defines the location and orientation of the component on the pcb. The parameters to this command include the X and Y coordinates of the device (in mils) and the device orientation. The orientation of the device is the rotation of the device with respect to the #1 pin or lead of the device. The following values are accepted by the placement command:

U (up) Pin 1 is towards the top of the board

D (down) Pin 1 is towards the bottom of the board

L (left) Pin 1 is towards the left edge of the board

R (right) Pin 1 is towards the right edge of the board

Anticipating and achieving the development of a routing algorithm that is fully aware of surface mounted devices, there is another parameter which denotes the side of the board on which the device is mounted. This parameter may have one of the following values:

C Component side

S Solder side

B Both sides (standard)

The form of this command is:

```
@ <component symbol> <X coord> <Y coord> <Rotation> <Side>
```

For example,

```

: place IC U1 at 1200,400 in a regular way
@ U1 1200 400 Up Both
:place edge connector P12 on the left edge
: pin 1 will be towards the top
@ P12 0 8000 Up Both

```

A.2.3 Signal Name

This command names the current signal that is being assigned to the various component pins and leads. The form of this command is:

```
! signal_name width-of-trace spacing-from-trace
```

The Auto-Board System II auto-router can accept various widths of traces and the spacing of other traces around that trace. You may specify the width of a signal on the signal name line. A value of 0 (zero) will cause the default value to be used (set by the configuration file). A second and optional value is for spacing between this signal and other traces. This value represents the edge to edge clearance in mils between all segments of this net and other traces on the board. If another signal has a larger spacing value, the larger is used during routing for spacing between the corresponding traces.

For example,

```

! DATA001
! DATA00F 10
! ADDR001 0 15

```

Pin assignments would appear between the signal name declarations since there can only be ONE current signal name.

In the case of simple and unnamed connections, it is unnecessary to invent nonsense names to fill the needs of this command. The netlist processor will provide signal names (in the form of a dollar sign and a number "\$12") when the signal name is a question mark ("?").

For example,

```
! ?
```

A.2.4 Signal Connection

This command assigns the current signal to a specific component pin or lead.

The netlist structure allows you to specify both the ganged pins of an array of integrated circuits and identity of pins that receive or generate independent signals. To specify a signal that is to be common to all packages in the array, simply name the array and the pin (i.e., A9 12 for pin 12 of array 9). To specify an independent signal and pin, name the particular package and the independent pin. (i.e., A3-4 6 is for pin 6 of the fourth package of array 3).

The command is initiated with an asterisk (*). The asterisk is followed by a single assignment or a list of assignments. They do not have to be on the same line. The first argument in the assignment is the package identifier (used in the part definition command). The second argument is the pin to which the current signal is to be connected. The form of this command is:

```
* <component symbol> <pin number>
```

For example:

```
! GND
```

```
* U1 8 U2 8 U3 8 U4 8
```

will connect pin #8 of devices U1 through U4 to ground (GND).

A.2.5 Device Type

This command labels a component with a device specification or value. In its present state, this command is used to document the specific name of part being used. The form of this command is:

```
# <component symbol> <device specification>
```

With this command, you can specify resistance values to resistors

```
# R1 270K
```

or device types to integrated circuits.

```
# U1 8086
```

A.2.6 Include File

This command allows you to combine several files in the definition of a pcb layout. Although several interesting uses for this command can be found, its main purpose is to support the formation of a library of standard declarations. In this way, common parts may be inserted into a design as required without having to reenter the specifics of the part for every project. This command is entered in this fashion:

+ file_name

The file is assumed to have the filetype of ".NET" and will be added by the netlist processor. You may have as many of these commands in a file as you wish. There is no limit. However, once a file inclusion command has been issued, a new level of file handling has been entered. When the process of reading the file has concluded, that level is left. Up to 10 levels of depth (number of netlist files initiating other netlist files) can be safely used. It is risky (and confusing) past that number. The example at the end of the chapter illustrates the use of this command.

A.2.7 Comment

The contents of the comment line are ignored by the netlist processor which allows you to document the contents of the netlist file. The form of the command is:

: explanatory text

Several examples have been shown in the commands above.

A.2.8 End of List

This command officially closes the processing of the netlist processor. The form of the command is:

Appendix B

Router Intermediate Programs

B.1 Introduction

The following sections describe the operation of the auto-router's intermediate programs. Between the netlist reader and the actual router are several programs that help organize the data structures written by the netlist reader. These programs are of little interest to the average user; but since you will be seeing them operate, you may be curious as to what they are doing.

The *ARRAY*, *EDGECON*, *BUSS* and *SCREEN* programs have user defined parameters which can control some of the operations of these programs. Modify the RTE batch stream if you want to invoke these options.

B.2 Array Processor

The Array program is activated by the presence of an array definition file, *.ARR* when the batch stream is used. This program is responsible for the organization of signals, traces and insertion of components for memory arrays. These arrays are declared in the netlist which is produced by the Layout program. The Array program provides the zig-zag trace patterns (in the script file) for connecting the common pins in a memory array.

The following line is the command line to the Array program. Don't attempt to run the program with this command. Use the batch files instead.

```
array test
```

There are two user defined parameters for the Array program. These are set in

the command line to the Array program.

Trace Width The width of the traces used in routing the common connections in the array can be set with this option. The default size is 15 mils. This option is set with the "-W" parameter on the command line. The "W" is immediately followed by the width of the trace. Using the value of zero will cause lines to be used. The following shows the width command in the batch file (set to 10 mils):

```
array %1 -w10
```

Polylines The Array program will generate AutoCAD polyline commands instead of lines or traces if the polyline parameter is present on the command line. The ADE-3 extension of AutoCAD must be present to use polylines. This option is set with the "-P" parameter on the command line. The following shows the polyline command in the batch file:

```
array %1 -p
```

B.3 Edge Connector

The Edge Connector program is activated by the presence of an edge connector definition file (.COM) when the batch file is used. This edge connector program is responsible for the organization of signals, pad placement and finger definition as declared in the netlist. The edge connector program provides the AutoCAD commands in the script file for the edge connector fingers and leads from the active routing area to those fingers. To keep the router from intruding into this defined area, an area mask is passed to the router to protect the predefined zone.

There are two parameters that should be used with this program (as appropriate). The first parameter is to let the program know if you are using the fineline router. The second parameter lets the program know whether you have surface mount routing capabilities.

The fineline parameter is the "2" option. It is entered on the command line for the edge connector program.

```
edgecon %1 -2
```

The surface mount parameter is the "S" option. It is entered on the command line for the edge connector program. The fineline router must be used with this option.

edgecon %1 -2 -S

This program comes in several configurations. See the installation notes in chapter 2 for more information.

B.4 Header

The Header program is activated by the presence of a header definition file (.HDR) when the batch stream is used. This header program is responsible for the organization of signals and pad placement of headers declared in the netlist. AutoCAD INSERT commands for 50 mil feedthrus are written to the script file for the creation of the headers.

This program comes in several configurations. See the installation notes in chapter 2 for more information.

B.5 Buss

The Buss program is activated in the batch file when the .BUS file exists for the current board. The Buss program is responsible for deciding where on a signal buss that a trace from a component should connect. Although the best decision is not always made here, the shortest distance between the buss and pin is calculated and that is the point of connection. Use AutoCAD to correct small deformities caused by this simple approach. However, the shortest distance (a straight line usually) insures an early routing of that trace.

In the case where pins of several packages are making their way to the buss, the Buss program will chain the signal from pin to pin within a narrow channel. This will cause power and ground pins of a package to be routed through a bypass capacitor (if present) instead of both sending tracks to the buss strip.

The BUSS program will generate polylines for the buss lines if the polyline parameter is set on the command line. This parameter is "-P".

For the fineline router, put an additional parameter on the command line to the buss processor in the batch streams. This parameter is "-F". It will take advantage of the trace spacing and variable width features of these advanced routers. You may find that this parameter has already been put into the command line for you.

B.6 Signal Sorting

The Sigsort program is always run the sequence of programs leading to the router. Sigsort is responsible for collecting all of the pins sharing the same signal into groups in preparation for signal chaining.

B.7 Signal Grouping

The Group program is responsible for the grouping and chaining of signals. The Group program calculates the various distances between pins of the same signal and schedules the routing pairs from this information. The chaining algorithm is not a simple chaining point to point system. The routing pairs will fan out or create hierarchical structures, depending upon the layout of the board.

B.8 Signal Screening

The Screen program is responsible for the ordering of the scheduled pairs of points to be routed. The highest priority for routing is the thickest traces (over some threshold). This allows the thicker traces for power and ground to be routed before the board becomes cluttered with all of the smaller signal traces. The next priority of traces is the simple connections. These include the "zero delta" traces that are perfectly vertical or horizontal.

The Screen program has some user defined parameters. These parameters modify the behavior of the program to meet some of your needs.

Trace Priority This parameter sets the width of trace that will take priority over the simple traces to be routed first. The value associated with this value is the width of the trace in mils. The "-W" command is used on the command line to set the threshold. The default is 20 mils. The following line shows the parameter as it might appear in the batch file.

```
screen %1 -w25
```

Reverse Priority This parameter reverses the priority rules for scheduling routes. With parameter in effect, the longer traces are routed first and the simpler and shorter ones last. This parameter does not change the priority of the wide traces. The "-X" command is used on the command line to set this parameter. The following line shows the parameter as it might appear in the batch file.

```
screen %1 -x
```

B.9 Prerouter

The Prerouter program can be used to assign critical traces to the layout of the board before the router attempts to route it. This feature allows you to enter power and ground tracks before they get mixed up in the normal routing chores.

The Prerouter should be activated through the PRTE batch file. This batch file operates in the same way as the RTE batch file except that it activates the prerouting sequence just before it starts the router.

The following list shows the basic steps needed to use the Prerouter.

- **AutoCAD** will draw the layout of the board as it appears before routing starts.
- Set the snap to 0.050 if using the grid router. It would be a good idea to set the snap to at least 0.010 for the high resolution routers to insure hitting the centers of pads.
- Set the layer to **SOLDER** or **COMPONENT** for the grid routers.
- Set the layer to **L1** or **L2** for the non multilayer routers.
- Set the layer to **L1** through **L16** for the multilayer router.
- Use the **LINE** or **PLINE** command to draw the prerouted traces. Stay on a single layer for laying out the initial tracks.
- Use the **CHANGE** command to change the lines on one layer to the other layer. For the grid routers, the **SOLDER** layer is predominately horizontal while the **COMPONENT** layer is predominately vertical.
- Write out the DXF file with the **DXFOUT** command.
- Exit the **AutoCAD** session with **QUIT**.
- The rest of the processing is automatic.

The Prerouter will tell the router to reserve the areas that it finds in the DXF file. It will also look through the pairs of scheduled to finding matching coordinates between the list of pairs and the endpoints of the lines drawn in **AutoCAD**. If there is a match, the scheduled pair is dropped from the routing schedule.

Appendix C

Symbol Libraries

TTL Library

Installation Guide

Perform the following steps to install the *TTL Library* disks into your Auto-Board System II.

- Place the disk marked as **TTL A** into the floppy drive A.
- Change the current directory to the *softwest* directory.

```
c:  
cd \softwest
```

- Copy the TTL.LIB, TTL.DTA and TTL.IDX files to the *softwest* directory.

```
copy a:ttl.*
```

- Add a reference line to the SOFTWEST.DIR file to activate the library. This line should be: **TTL 2**. Use the following steps to do this:

```
EDLIN SOFTWEST.DIR  
I  
TTL          2  
ctrl-c  
E
```

- Copy the TTL library drawings into the schematic capture library.
 1. Change the current directory to the schematic capture directory.

```
cd \sc
```
 2. Place the disk marked as **TTL 2** into the floppy drive A.
 3. Copy the drawing files from the disk.

TTL Library

Installation Guide

Perform the following steps to install the TTL Library disks into your Auto-Board System II.

- Place the disk marked as TTL A into the floppy drive A.
- Change the current directory to the software directory.

```
cd /software
```

- Copy the TTL.LIB, TTL.DTA and TTL.IDX files to the software directory.

```
copy a:ttl *
```

- Add a reference line to the SOFTWARE.DIR file to activate the library. This line should be: TTL 3. Use the following steps to do this:

```
EDLIN SOFTWARE.DIR
I
TTL 3
ctrl-c
E
```

- Copy the TTL library drawings into the schematic capture library.
- 1. Change the current directory to the schematic capture directory.

```
cd /sc
```

- 2. Place the disk marked as TTL 3 into the floppy drive A.
- 3. Copy the drawing files from the disk.

copy a:*.dwg

4. Place the disk marked as **TTL 3** into the floppy drive A.
5. Copy the drawing files from the disk.

copy a:*.dwg

When this AutoCAD menu command is used, you will be prompted for the drawing name and the insert point. Use the name of the one of the parts in the library. That part will be inserted at the location of the cursor. You will be prompted for a reference designator. This must be entered since this is the name that ties the schematic capture to the router.

- The script files (*.SCR) are on disk **TTL 1**. These are for your convenience in altering any of the drawings as you may desire. Refer to the user guide for The Auto-Board System (chapters 6 and 8) for more information about custom libraries.
- This completes the installation of the *TTL Library*.

28 Aug. 1986

ERRATA NOTES:

1. There should have been a file by the name of "OLD.SCR" on the LAYOUT disk. However, it would have been an exact copy of the file by the same name on the SCHEMATIC CAPTURE disks. Please copy this file to the LAYOUT directory.

COPY \SC\OLD.SCR \LAYOUT\OLD.SCR

2. We have had reports recently on the ROUTE50 50 mil grid router that using the PREROUTE with the FEEDTHRU REDUCTION optimizer turned ON will cause the program to hang up. These reports are for the version 1.3a, but since there is little change in the programs in this area, we would suspect that this problem still exists in the 50 mil router. When we track down the problem, it will be fixed and added to a future update. Sorry.
3. For the strong of heart, there is an undocumented feature that was added at the last minute. In the router configuration file, the parameter "PULL" will cause traces to the middle of feedthrus and pads to be pulled back so that you can see the board through the hole of a donut pad. To pull back 10 mils from the center, the parameter line would be:

PULL = 10

4. In the RTE and RTE50 batch files, towards the end of the files, you may find a command that reads:

IF EXIST %1.PAD DEL %1.PRE

this should read:

IF EXIST %1.PRE DEL %1.PRE

This can be fixed with a text editor or word processor.

5. When using the prerouter feature, don't allow corners of two tracks appear on opposite sides of the board at the same x,y coords. They will get picked up as the same track; which is what is supposed to happen if you want to change sides in the prerouter.

Dennis


```
echo off
if exist %1.dwg del %1.dwg
if exist %1.scr del %1.scr
net %1 %1-L %1-S
if exist %1.hdr header %1
if exist %1.hdr del %1.hdr
if exist %1.arr array %1 -w10 -p
if exist %1.arr del %1.arr
if exist %1.con edgecon %1 -2
if exist %1.con del %1.con
if exist %1.bus buss %1
if exist %1.bus del %1.bus
group %1
del %1.grp
route %1
cls
echo Loading AutoCAD
if exist %1.pad del %1.pad
if exist %1.def del %1.def
if exist %1.edg del %1.edg
if exist %1.pre del %1.pre
if exist %1.000 del %1.000
if exist %1.1 del %1.1
if exist %1.2 del %1.2
if exist %1.3 del %1.3
if exist %1.4 del %1.4
if exist %1.5 del %1.5
if exist %1.6 del %1.6
set acad=\router
acad %1 %1
```

RTE

KTSO

```
echo off
if exist %1.dwg del %1.dwg
if exist %1.scr del %1.scr
net %1 %1-L %1-S
if exist %1.hdr header %1
if exist %1.hdr del %1.hdr
if exist %1.arr array %1 -w10 -p
if exist %1.arr del %1.arr
if exist %1.con edgecon %1
if exist %1.con del %1.con
if exist %1.bus buss %1
if exist %1.bus del %1.bus
group %1
del %1.grp
route50 %1
cls
echo Loading AutoCAD
if exist %1.pad del %1.pad
if exist %1.def del %1.def
if exist %1.edg del %1.edg
if exist %1.pre del %1.pre
if exist %1.000 del %1.000
if exist %1.1 del %1.1
if exist %1.2 del %1.2
if exist %1.3 del %1.3
if exist %1.4 del %1.4
if exist %1.5 del %1.5
if exist %1.6 del %1.6
set acad=\router
acad %1 %1
```


PRTE.

```
echo off
if exist %1.dwg del %1.dwg
if exist %1.scr del %1.scr
net %1 %1-L %1-S
if exist %1.hdr header %1
if exist %1.hdr del %1.hdr
if exist %1.arr array %1 -w10 -p
if exist %1.arr del %1.arr
if exist %1.con edgecon %1
if exist %1.con %1.con
if exist %1.bus buss %1
if exist %1.bus del %1.bus
group %1
del %1.grp
set acad=\router
acad %1 %1
preroute %1
del %1.def
ren %1.tmp %1.def
route50 %1
cls
echo Loading AutoCAD
if exist %1.pad del %1.pad
if exist %1.def del %1.def
if exist %1.edg del %1.edg
if exist %1.pre del %1.pre
if exist %1.000 del %1.000
if exist %1.1 del %1.1
if exist %1.2 del %1.2
if exist %1.3 del %1.3
if exist %1.4 del %1.4
if exist %1.5 del %1.5
if exist %1.6 del %1.6
set acad=\router
acad %1 %1
```

8415.

```
echo off
if exist %1.dwg del %1.dwg
if exist %1.scr del %1.scr
del %1 %1-L %1-S
if exist %1.hdr header %1
if exist %1.hdr del %1.hdr
if exist %1.scr array %1 -w10 -p
if exist %1.scr del %1.scr
if exist %1 con edgcon %1
if exist %1 con %1 con
if exist %1.pus pus %1
if exist %1.pus del %1.pus
group %1
del %1.grp
set acad=/router
acad %1 %1
preroute %1
del %1.def
ren %1 %1.def
route50 %1
cls
echo Loading AutoCAD
if exist %1.pas del %1.pas
if exist %1.def del %1.def
if exist %1.edg del %1.edg
if exist %1.pre del %1.pre
if exist %1.000 del %1.000
if exist %1.1 del %1.1
if exist %1.2 del %1.2
if exist %1.3 del %1.3
if exist %1.4 del %1.4
if exist %1.5 del %1.5
if exist %1.6 del %1.6
set acad=/router
acad %1 %1
```

```
echo off
if exist %1.dwg del %1.dwg
if exist %1.scr del %1.scr
net %1 %1-L %1-S
if exist %1.hdr header %1
if exist %1.hdr del %1.hdr
if exist %1.arr array %1 -w10 -p
if exist %1.arr del %1.arr
if exist %1.con edgecon %1
if exist %1.con %1.con
if exist %1.bus buss %1
if exist %1.bus del %1.bus
group %1
del %1.grp
set acad=\router
acad %1 %1
preroute %1
del %1.def
ren %1.tmp %1.def
route50 %1
cls
echo Loading AutoCAD
if exist %1.pad del %1.pad
if exist %1.def del %1.def
if exist %1.edg del %1.edg
if exist %1.pre del %1.pre
if exist %1.000 del %1.000
if exist %1.1 del %1.1
if exist %1.2 del %1.2
if exist %1.3 del %1.3
if exist %1.4 del %1.4
if exist %1.5 del %1.5
if exist %1.6 del %1.6
set acad=\router
acad %1 %1
```

PAGE 50

12/27

```
echo off
if exist %1 dwg del %1 dwg
if exist %1 acb del %1 acb
del %1 %1-2
if exist %1 hbr header %1
if exist %1 hbr del %1 hbr
if exist %1 arr array %1 -w10 -p
if exist %1 arr del %1 arr
if exist %1 con edgecon %1
if exist %1 con %1 con
if exist %1 pus pus %1
if exist %1 pus del %1 pus
group %1
del %1 grp
set acad=router
acad %1 %1
proute %1
del %1 dat
ren %1 tmp %1 def
route50 %1
cls
echo Loading AutoCAD
if exist %1 pad del %1 pad
if exist %1 def del %1 def
if exist %1 edg del %1 edg
if exist %1 pre del %1 pre
if exist %1 000 del %1 000
if exist %1 1 del %1 1
if exist %1 2 del %1 2
if exist %1 3 del %1 3
if exist %1 4 del %1 4
if exist %1 5 del %1 5
if exist %1 6 del %1 6
set acad=router
acad %1 %1
```

4 Feb 1987

ERRATA NOTES:

1. (FIXED WITH THIS RELEASE -- DISREGARD) There should have been a file by the name of "OLD.SCR" on the LAYOUT disk. However, it would have been an exact copy of the file by the same name on the SCHEMATIC CAPTURE disks. Please copy this file to the LAYOUT directory.

COPY \SC\OLD.SCR \LAYOUT\OLD.SCR

2. We have had reports recently on the ROUTE50 50 mil grid router that using the PREROUTE with the FEEDTHRU REDUCTION optimizer turned ON will cause the program to hang up. These reports are for the version 1.3a, but since there is little change in the programs in this area, we would suspect that this problem still exists in the 50 mil router. When we track down the problem, it will be fixed and added to a future update. Sorry.

3. For the strong of heart, there is an undocumented feature that was added at the last minute. In the router configuration file, the parameter "PULL" will cause traces to the middle of feedthrus and pads to be pulled back so that you can see the board through the hole of a donut pad. To pull back 10 mils from the center, the parameter line would be:

PULL = 10

4. (FIXED WITH THIS RELEASE -- DISREGARD)
In the RTE and RTE50 batch files, towards the end of the files, you may find a command that reads:

IF EXIST %1.PAD DEL %1.PRE

this should read:

IF EXIST %1.PRE DEL %1.PRE

This can be fixed with a text editor or word processor.

5. When using the prerouter feature, don't allow corners of two tracks appear on opposite sides of the board at the same x,y coords. They will get picked up as the same track; which is what is supposed to happen if you want to change sides in the prerouter.
6. The SIGSORT and SCREEN programs have been incorporated within the GROUP program. Disregard any mention of them in the user's guide. The GROUP program has a new algorithm and is about a zillion times faster than the 1.3 and 2.0 version.
7. The NETGEN program will produce a SCRIPT file to draw red circles at questionable (not necessarily wrong) places in a schematic. This feature can be disabled with the "-w" parameter on the command line.
8. Schematic capture recognizes a connection between symbols when their package leads are touching. In the past, a line had to be drawn between package leads to show connectivity. A red warning circle may be displayed at such a point, but you can ignore its significance in this instance.
9. The EDGECON and HEADER programs will INSERT a HDR and EDGEPT block in your routed drawing instead of drawing the parts as in the past. This is done

653

16.75.42

432 23
P5 42

for compatibility with the board verification programs in the tool kit.

10. A Gerber interface program is now available.

11. NETBILD will create the NETLIB.RT file automatically now, you do not have to edit this file when building the router/netlist database.

12. There was a bug in LAYOUT causing it to generate meaningless # netlist commands. This has been fixed.

13. If you use Wordstar to edit the netlists, make sure that there is a period at the end of the file. This indicates a fake end-of-file to the netlist reader. Wordstar pads the end of the file with strange characters that hang the netlist reader.

14. In the fineline router, a new option has been added. You can change the resolution of the router from 1 mil to just about anything that you would like. When you change the scale, the size of the routing area increases. With 1 mil, the board is 16" by 16". At a 2 mil resolution, the board is 32" by 32". At 5 mil, you get 80" by 80". Set the resolution with the GRID command.

GRID = 2

in the configuration file will set the router to 2 mil resolution.

GRID = 5 is 5 mil resolution.

Co-ordinates and pad sizes will be TRUNCATED to the proper resolution. An 8 mil trace in 5 mil resolution will become 5 mils. A 13 mil trace in 2 mil resolution will become 12 mils. Try not to pick resolutions that will cause numbers to become zero.

15. We are thinking of dropping support for the TRACE script command in the next release. The internal representation of TRACE given by AutoCAD is strange and the Gerber interface is not very fond of them ... and I don't know of anybody using them anyway.

16. In the schematic drawing, connecting a number of components with a looped circuit will cause the NETGEN program to hang or run out of stack. If you get a stack overflow, check the xxx-S.NET file. The last entry in the file should be one of the members in the loop. Correct the schematic to break the loop.

17. Placing components, board edges, schematic symbols, ANYTHING into a negative coordinate (out of the first quadrant) can cause disastrous results. Check this condition before calling.

18. Editing the board edges in the LAYOUT portion can cause strange edge definitions to be written to the netlist. Edges should be drawn as one continuous sequence of lines. We'll be fixing this, but it isn't yet.

19. In the user guide, Chapter 7 discusses the attributes for LAYOUT parts. One of the attributes is called NAME in the guide. This attribute should have been called PART. This shows up a couple of times.

20. The schematic netlist generator now includes the name of the drawing from which the netlist was extracted for each part number / pin number pair. This will prove to be handy when several netlists are combined and sorted in the netlist pretty printer in the tool kit.